Chuyên đề ngôn ngữ lập trình AutoIt

AutoIt là một ngôn ngữ lập trình có cú pháp tương tự gần như ngôn ngữ BASIC, được cung cấp miễn phí cho các lập trình viên. Ngôn ngữ lập trình AutoIt có công dụng để thực hiện một số thao tác tự động hóa như giả lập phím nhấn, di chuyển chuột, và thực hiện một số thao tác phức tạp khác trên cửa sổ, thao tác với tập tin, thư mục... Tất cả các chức năng kể trên của ngôn ngữ AutoIt có được là nhờ những hàm chức năng của nó. Nội dung cuốn sách xin trình bày một số hàm hữu ích của AutoIt như các hàm liên quan đến xử lý bàn phím, chuột, thao tác trên cửa sổ cũng như các hàm liên quan đến xử lý tiến trình, Registry,... Đồng thời sẽ có phần hướng dẫn sử dụng công cụ Koda FormDesigner để thiết kế giao diện trong AutoIt.

Hàm xử lý biến môi trường trong Windows

Thông thường để thiết lập biến môi trường trong Windows, bạn hay bấm phải chuột vào My Computer > Properties >Advanced, chọn Environment Variables (Windows XP). Bây giờ bạn có thể tận dụng một số hàm có sẵn xử lý biến môi trường của AutoIt để làm việc này mà không phải làm thao tác trên. Trong phần này cũng xin giới thiệu thêm một số hàm liên quan đến Clipboard, Memory.

1. EnvGet ("Envvar")

- Công dụng : lấy thông tin của một biến môi trường trong Windows.

- Envvar là tên biến môi trường cần lấy thông tin về. Ví dụ như "TEMP" hoặc "PATH".

Ví dụ : *$tam = EnvGet("TEMP")*; lấy thông tin biến môi trường TEMP đưa vào biến $tam.

MsgBox(4096, "Thong tin bien moi truong TEMP la : ", $tam); xuất hiện hộp thoại thông báo kết quả của hàm.

2. EnvSet("Envvar", ["value"])

- Công dụng : ghi dữ liệu cho một biến môi trường trong Windows.

- Envvar là tên biến môi trường cần ghi dữ liệu. value là một tham số tùy chọn, nó chứa thông tin dữ liệu cần ghi vào biến môi trường. Nếu tham số này không có trong hàm, thì biến môi trường "Envvar" sẽ bị xóa khỏi Windows.

Ví dụ : *EnvSet("path", "D:\TCWIN45\BIN")*

3. EnvUpdate()

- Công dụng : refresh lại biến môi trường của Windows. Thông thường khi bạn thực hiện một số thao tác cập nhật biến môi trường trong Windows thì bắt buộc phải khởi động lại máy mới có hiệu lực. Nay bạn chỉ cần gọi hàm này ngay sau khi thực hiện cập nhật biến môi trường mà không cần khởi động lại máy để hoàn thành cấu hình.

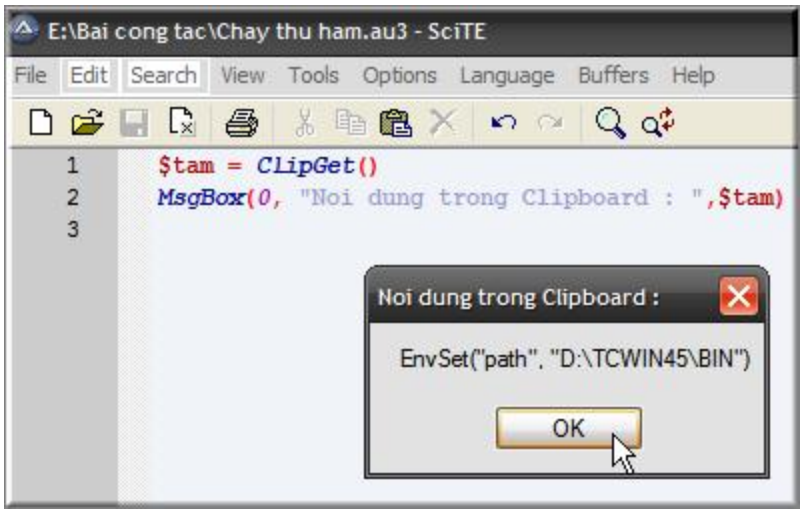
- Ví dụ : *EnvSet("path", "D:\TCWIN45\BIN")*

EnvUpdate()

4. ClipGet()

- Công dụng : lấy thông tin văn bản được chứa trong Clipboard

- Ví dụ :



5. ClipPut("text")

- Công dụng : ghi dữ liệu văn bản vào Clipboard.

- Tham số text trong hàm là dữ liệu văn bản cần ghi.

Ví dụ : *ClipPut("Mr. Trinh – DH10TH")*

6. MemGetStats()

- Công dụng : trả về thông tin bộ nhớ máy tính của bạn. Dữ liệu trả về sẽ chứa trong một mảng có bảy phần tử. Trong đó phần tử đầu tiền chứa thông tin tỉ lệ phần trăm bộ nhớ đã dùng. Phần tử thứ hai chứa tổng dung lượng bộ nhớ vật lý. Phần tử thứ ba chứa thông tin bộ nhớ vật lý có sẵn để sử dụng. Phần tử thứ tư chứa thông tin tổng dung lượng Pagefile. Phần tử thứ năm chứa thông tin dung lượng Pagefile còn lại có sẵn để sử dụng. Phần tử thứ sáu chứa thông tin dung lượng bộ nhớ ảo. Phần tử thứ bảy chứa thông tin dung lượng bộ nhớ ảo còn lại có sẵn có thể sử dụng. Đơn vị dung lượng bộ nhớ tính bằng KB.

Ví dụ : *$tam = MemGetStats()*

MsgBox(0, "Phan tram dung luong bo nho da dung ", $tam[0])

Hàm quản lý tập tin, thư mục và phân vùng của đĩa cứng

1. DirCopy("sourcedir", "destdir", [flag])

- Công dụng : sao chép toàn bộ nội dung bao gồm cả thư mục con và tập tin của thư mục nguồn đến thư mục đích.

- Tham số sourcedir chỉ thư mục nguồn cần sao chép. destdir chỉ thư mục đích mà thư mục nguồn được sao chép tới. flag là tham số tùy chọn trong hàm này, nếu flag = 0 thì sẽ không ghi đè tập tin nếu nó tồn tại sẵn ở thư mục đích, ngược lại nếu flag = 1 sẽ ghi đè tập tin ở thư mục đích nếu nó trùng với tập tin ở thư mục nguồn. Tham số flag = 0 được bật mặc định. Nếu thư mục đích không tồn tại, hàm sẽ tạo thư mục đích cho bạn.

Ví dụ : *DirCopy(@ProgramFilesDir, "D:\BackupPro", 1) ;* sao chép nội thư mục Program Files trong phân vùng chứa hệ điều hành Windows sang thư mục D:\BackupPro.

2. DirCreate("path")

- Công dụng : tạo thư mục.

- Tham số path là đường dẫn bạn muốn tạo thư mục.

Ví dụ : *DirCreate("D:\Document\Cam nang")* ; tạo thư mục Cam nang trong thư mục Document. Trong trường hợp thư mục Document không tồn tại, hàm cũng sẽ tạo sẵn cho bạn.

3. DirGetSize("path", [flag])

- Công dụng : trả về kích thước một thư mục. Đơn vị tính : Bytes.

- Tham số path là đường dẫn bạn cần lấy kích thước thư mục. flag là tham số tùy chọn, nếu flag = 1, hàm này sẽ trả về một mảng có ba phần tử. Trong đó phần tử thứ nhất chứa dung lượng của thư mục, phần tử thứ hai chứa số tập tin, phần tử thứ ba chứa số thư mục con. Nếu flag = 2, hàm này chỉ trả về dung lượng của thư mục mà thôi.

Ví dụ : *$size = DirGetSize("D:\Office", 2)*

MsgBox(0, "SizeOffice", "Kich thuoc thu muc la "&$size)

4. DirMove("sourcedir", "destdir", [flag])

Hàm này có tham số và cách sử dụng tương tự như hàm DirCopy, nhưng chỉ có khác đây là di chuyển nội dung bên trong một thư mục sang thư mục khác.

5. DirRemove("path", [flag])

- Công dụng : xóa một thư mục.

- Tham số path là đường dẫn của thư mục cần muốn xóa. flag là tham số tùy chọn, mặc định là 0. Trường hợp flag = 0, nếu thư mục cần xóa có thư mục con hoặc tập tin thì sẽ không xóa được. Trường hợp flag = 1, nếu thư mục cần xóa có thư mục con và tập tin thì sẽ xóa nó luôn.

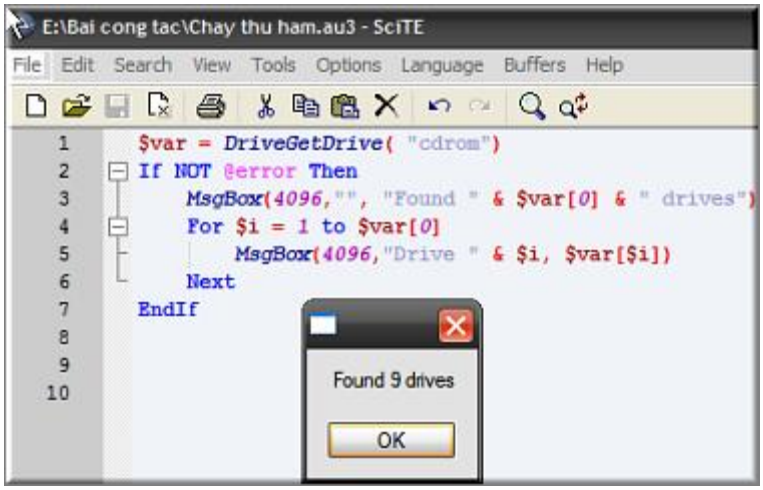
Ví dụ : *DirRemove("D:\Thu vien sach noi", 1)*

6. DriveGetDrive("Type")

- Công dụng : trả về các kí tự ổ đĩa trong máy tính mà nó tìm thấy.

- Tham số Type chỉ loại thiết bị mà bạn muốn lấy về. Hàm sẽ trả về một mảng, trong đó phần tử đầu tiên là số ổ đĩa hay thiết bị mà nó tìm thấy. Các phần tử còn lại trong mảng sẽ lần lượt chứa các kí tự ổ đĩa hay thiết bị. Các loại thiết bị có thể là : CDROM, NETWORK, REMOVABLE, RAMDISK, UNKNOWN, ALL (tất cả các thiết bị trên).

- Ví dụ :



Trong máy tính của tôi hiện có 9 ổ đĩa CDROM (1 ổ đĩa vật lý, 8 ổ đĩa ảo). Do đó sẽ có hộp thoại thông báo như trên. Sau đó nó sẽ liệt kê các ổ đĩa CDROM trong máy tính của tôi.

7. DriveGetFileSystem ("path")

- Công dụng : trả về hệ thống tập tin của phân vùng ổ đĩa nào đó của bạn.

- path là tham số đường dẫn ổ đĩa của bạn. Hàm này sẽ trả về một giá trị chuỗi. Nếu giá trị trả về là 1, ổ đĩa của bạn không được định dạng hoặc hàm không nhận ra. Ngoài ra nó sẽ trả về các giá trị như FAT, FAT32, NTFS, NWFS (hệ thống tập tin Novell Netware File Server), CDFS, UDF.

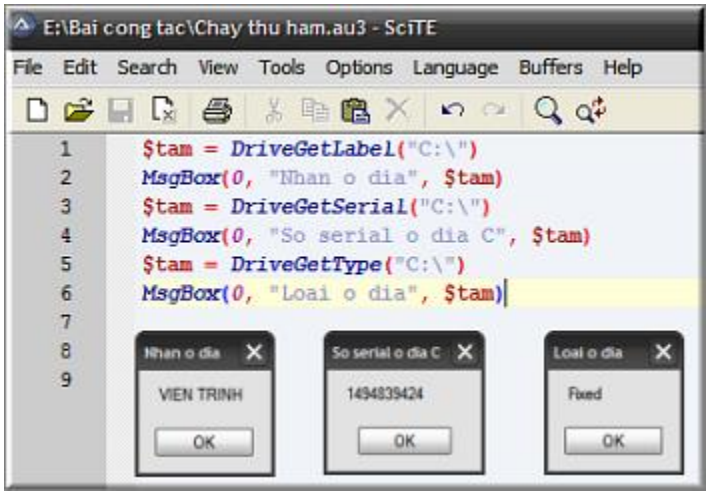
- Ví dụ : *$tam = DriveGetFileSystem("D:\")*

MsgBox(4096,"File System Type:", $tam)

8. DriveGetLabel("path"), DriveGetSerial("path"), DriveGetType("path"), DriveSpaceFree ("path"), DriveSpaceTotal("path").

Các hàm trên có công dụng lần lượt là lấy nhãn của ổ đĩa, lấy số serial, lấy thông tin loại ổ đia, trả về khoảng không gian còn trống trong ổ đĩa (MB), trả về tổng không gian trong ổ đĩa (MB). Trong đó path là đường dẫn đến phân vùng ổ đĩa của bạn.

Ví dụ :



9. DriveSetLabel("path", "label")

- Công dụng : thay đổi nhãn ổ đĩa của bạn.

- path : chỉ đường dẫn phân vùng ổ đĩa của bạn. label là tên nhãn cần gán cho ổ đĩa.

Ví dụ : *DriveSetLabel("D:\", "Mr. Trinh")*

10. DriveMapAdd("device", "remote share", [flags], ["user"], ["password"])

- Công dụng : thực hiện ánh xạ ổ đĩa tương tự như tính năng Map Network Drive trong Windows.

- Tham số device là kí tự ổ đĩa mà bạn muốn gán, remote share là đường dẫn đến thư mục ổ đĩa mạng mà bạn cần ánh xạ, flag : cờ trạng thái, nếu = 0 sẽ sử dụng tham số đăng nhập tự động mà bạn cung cấp và tiến hành kết nối liên tục, nếu = 1 sẽ xuất hiện hộp thoại yêu cầu đăng nhập username và password mỗi khi có yêu cầu từ thư mục mạng chia sẻ. user và password là thông tin tài khoản mà bạn cần cung cấp nếu như ổ đĩa ánh xạ của bạn phải có tài khoản để đăng nhập.

Ví dụ : *DriveMapAdd("S:", "\\NETWORKDRIVE\SHARE", 0, "mrvientrinh", "lvt*tl91")*

11. DriveMapDel("device")

- Công dụng : xóa kết nối ánh xạ ổ đĩa tương tự như tính năng Disconnect Network Drive

- device : là tham số tên ổ đĩa ánh xạ mà bạn cần xóa

Ví dụ : *DriveMapAdd("S:", "\\NETWORKDRIVE\SHARE", 0, "mrvientrinh", "lvt*tl91")*

DriveMapDel("S:")

12. DriveMapGet("device")

- Công dụng : xem thông tin kết nối của một ổ đĩa ánh xạ.

- Device : kí tự ổ đĩa ánh xạ mà bạn cần xem thông tin.

Ví dụ : *DriveMapAdd("S:", "\\NETWORKDRIVE\SHARE", 0, "mrvientrinh", "lvt*tl91")*

MsgBox(0, "Thong tin ket noi", DriveMapGet("S:"))

13. FileGetAttrib("filename")

- Công dụng : lấy thuộc tính của tập tin.

- filename : đường dẫn tập tin cần lấy thông tin thuộc tính. Hàm này sẽ trả về một chuỗi kí tự kết hợp đại diện cho các thuộc tính trông giống như "RASHNDOCT". Trong đó R : chỉ đọc, A : lưu trữ, S : hệ thống, H : ẩn, N : thông thường, D : thư mục, O : ngoại tuyến, C : nén, theo phân vùng có định dạng NTFS, không phải định dạng nén ZIP, T : thuộc tính tạm.

Ví dụ : *$thuoctinh = FileGetAttrib("C:\boot.ini")*

MsgBox(0, "Thuoc tinh cua tap tin boot.ini", $thuoctinh) ; kết quả "SH"

14. FileSetAttrib("filename", "attrib", [flag])

- Công dụng : Thiết lập một số thuộc tính cho tập tin.

- filename : tên tập tin bạn cần gán thuộc tính, bạn có thể dùng dấu * để miêu tả cho tập hợp các tập tin. attrib là thuộc tính bạn cần gán (bạn xem thuộc tính ở hàm FileGetAttrib). + : gán thuộc tính, - : bỏ thuộc tính. flag là tham số tùy chọn (áp dụng nếu bạn đang thiết lập thuộc tính cho một thư mục), nếu flag = 1 thì trong trường hợp bạn đang gán thuộc tính cho một thư mục thì toàn bộ thư mục con và tập tin nằm trong

thư mục đó đều có thuộc tính giống như thư mục đang thiết lập. Nếu flag = 0 (tham số mặc định), thì chỉ có thư mục nằm trong đường dẫn filename là gán thuộc tính mà thôi.

Ví dụ : *FileSetAttrib("D:*.*", "+S+H-R")* ; gán thuộc tính cho tất cả tập tin gốc trong ổ đĩa D thành tập tin hệ thống, ẩn nhưng bỏ thuộc tính chỉ đọc.

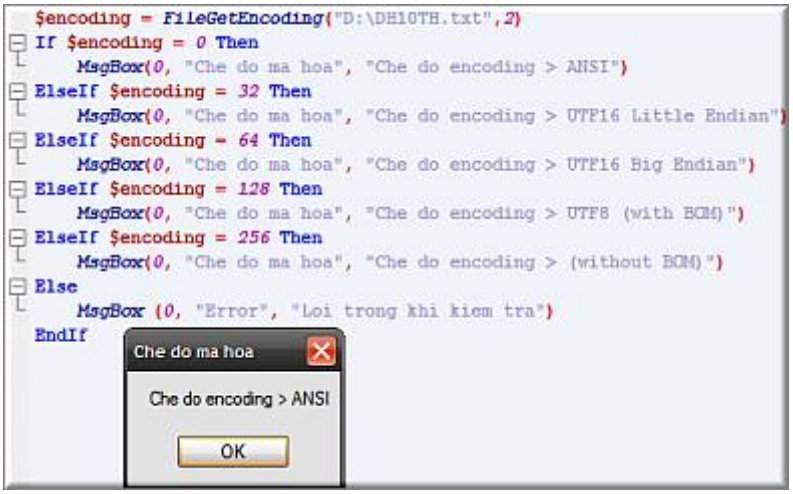
15. FileGetEncoding("filename", [mode])

- Công dụng : xác định chế độ encoding được dùng trong tập tin văn bản.

- filename : đường dẫn tập tin cần xác định, [mode]: tham số tùy chọn, mặc định là 1. Trong trường hợp mode = 1, hàm sẽ kiểm tra chế độ mã hóa toàn bộ tập tin để xác định. Trong trường hợp mode = 2, hàm sẽ kiểm trả chế độ mã hóa ở phần đầu tập tin. Nếu hàm gọi bị lỗi sẽ trả về giá trị -1. Nếu được gọi thành công, hàm sẽ trả về các giá trị như sau :

- 0 : tập tin được mã hóa theo ANSI.
- 32 : UTF16 Little Endian.
- 64 : UTF16 Big Endian.
- 128 : UTF8 (with BOM).
- 256 : (without BOM).

Ví dụ :



16. FileGetShortcut("lnk")

- Công dụng : lấy thông tin chi tiết về một shortcut nào đó.

- lnk là tham số đường dẫn chỉ đến shortcut mà bạn muốn xem thông tin. Hàm này sẽ trả về một mảng gồm bảy phần tử. Trong đó phần tử thứ nhất chứa đường dẫn của tập tin. Phần tử thứ hai chứa thông tin đường dẫn thư mục của tập tin. Phần tử thứ ba chứa các đối số của shortcut (nếu có). Phần tử thứ tư chứa thông tin mô tả cho shortcut. Phần tử thứ năm chứa đường dẫn của tập tin icon. Phần tử thứ sau chứa số chỉ mục của icon cho shortcut. Phần tử thứ bảy chứa thông tin kiểu của cửa sổ thực thi khi nhấp đôi vào shortcut bao gồm : 1 (cửa sổ hiện thông thường), 7 (cửa sổ thu nhỏ dưới thanh Taskbar), 3 (phóng lớn toàn cửa sổ chương trình).

Ví dụ : *$details = FileGetShortcut("C:\Documents and Settings\Mr. Trinh\Start Menu\Programs\Windows Media Player.lnk")*

MsgBox(0, "Duong dan cua Windows Media Player:", $details[0])

17. FileMove("source", "dest", [flag])

- Công dụng : di chuyển tập tin

- source đường dẫn tập tin nguồn cần di chuyển. dest : đường dẫn của thư mục đích mà tập tin cần di chuyển tới. flag là tham số tùy chọn, flag = 0 -> tiến hành ghi đè tập tin nếu thư mục đích có tồn tại tập tin đó, flag = 1 -> không ghi đè tập tin nếu nó tồn tại ở thư mục đích, flag = 8 > trong trường hợp đường dẫn thư mục đích bạn gõ vào không có thực, thì tham số này cho phép hàm tiến hành tạo cấu trúc thư mục đích. Bạn có thể kết hợp các tham số cờ flag trên bằng cách gõ số tổng của chúng. Ví dụ bạn muốn có tùy chọn cờ flag vừa là 1 và 8, thì bạn gán tham số flag = 9.

Ví dụ : *FileMove("D:*.txt", @TempDir & "\TxtFiles\", 9)*

18. FileOpenDialog("title", "dir", "filter", [options], ["defaultname"])

- Công dụng : mở hộp thoại Open. (thường được ứng dụng trong lập trình giao diện).

- title : là tên tiêu đề của cửa sổ Open mà bạn cần xác lập. dir là thư mục mặc định mỗi khi mở hộp thoại Open. filter là bộ lọc tập tin (chỉ cho phép hiển thị các tập tin trong một thư mục dạng đặc biệt nào đó khi chọn). options là tham số tùy chọn của hàm, có 4 tham số tùy chọn

1 > Tên tập tin được mở cần phải tồn tại (áp dụng cho người dùng mở tập tin bằng cách gõ đường dẫn).

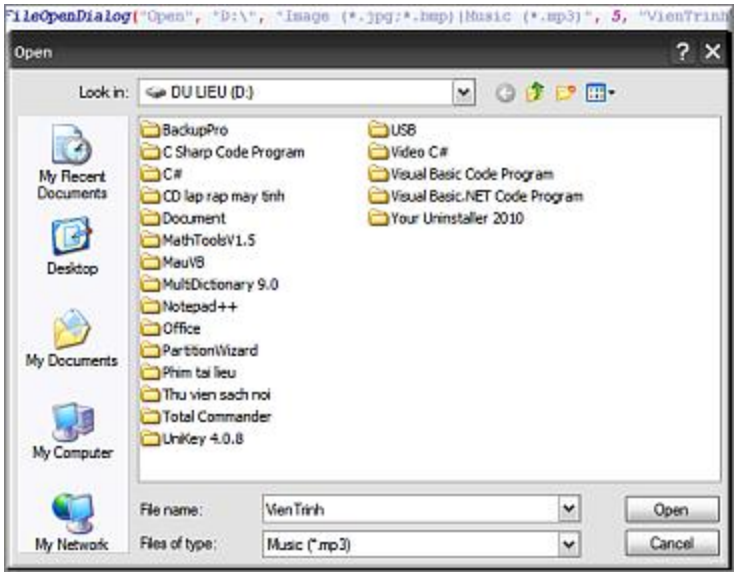
2 > Tên thư mục cần mở phải tồn tại (áp dụng cho người dùng mở thư mục bằng cách gõ đường dẫn).

4 > Cho phép chọn nhiều tập tin

8 > Xác nhận tạo tập tin mới nếu nó không tồn tại.

defaultname : là tên tập tin ngầm định nằm trong ô File name của hộp thoại Open. Hàm này nếu gọi thành công sẽ trả về đường dẫn đầy đủ của tập tin được chọn. Nếu chọn nhiều tập tin kết quả trả về sẽ có dạng như "Directory|file1|file2|...".

Ví dụ : *FileOpenDialog("Open", "D:\", "Image (*.jpg;*.bmp)|Music (*.mp3)", 5, "VienTrinh")*; tạo hộp thoại Open với tham số tên tiêu đề cửa sổ là Open, thư mục mặc định ổ D, loại tập tin cần mở là *.jpg, *.bmp, *.mp3, cho phép chọn nhiều tập tin và nếu người dùng gõ đường dẫn cho tập tin cần mở thì tập tin phải tồn tại (tham số options = 5, tổng của 1 và 4), tên tập tin ngầm định khi mở là VienTrinh.



19. FileRecycle("source")

- Công dụng : xóa tập tin và đưa vào thùng rác.

- source là đường dẫn của tập tin.

Ví dụ : *FileRecycle("D:*.mp3")*

20. FileRecycleEmpty(["rootpath"])

- Công dụng : xóa sạch tập tin trong thùng rác.

- rootpath : tham số tùy chọn. Nó là ổ đĩa gốc cho những tập tin mà bạn muốn xóa trong thùng rác. Nếu tham số này rỗng, sẽ xóa tất cả tập tin trong thùng rác.

Ví dụ : nếu bạn muốn xóa những tập tin trong thùng rác, nhưng chỉ xóa những tập tin có đường dẫn từ ổ D. Bạn gõ hàm sau : *FileRecycleEmpty("D:\")*.

21. FileSaveDialog("title", "dir", "filter", [options], ["defaultname"])

- Công dụng : mở hộp thoại Save.

- Các tham số tương tự như hàm FileOpenDialog, nhưng là thao tác Save. Tham số options cũng có khác đối chút, nó chỉ có hai tùy chọn, đó là :

2 > Đường dẫn thư mục cần phải tồn tại, nếu người dùng gõ đường dẫn lưu bằng tay.

16 > Xác nhận việc ghi đè tập tin nếu có trùng tên tập tin.

22. FileSelectFolder("dialogtext", "rootdir", [flag], ["dir"])

- Công dụng : mở hộp thoại chọn thư mục.

- dialogtext : dòng chữ giới thiệu sẽ xuất hiện trên cây thư mục. rootdir : thư mục gốc cho cây thư mục (nếu tham số này là "", thư mục gốc là Desktop), flag > cờ tham số tùy chọn, có ba tham số như sau :

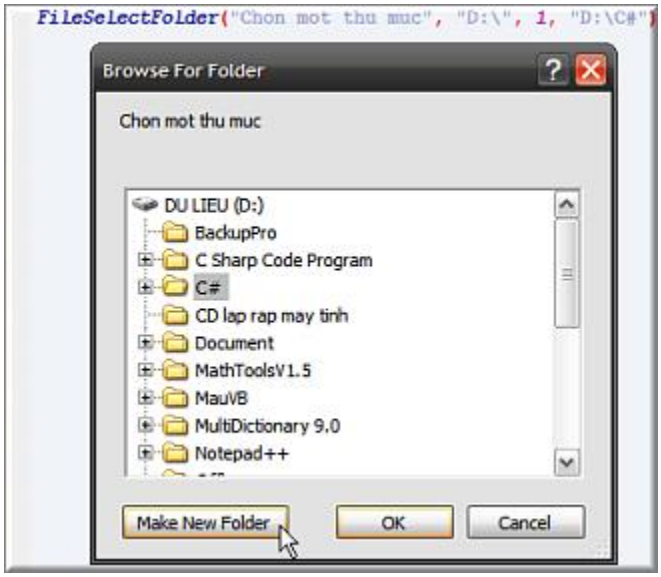
1 > Hiện nút Make New Folder trong hộp thoại chọn Folder

2 > Dùng style New Dialog cho hộp thoại Select.

4 > Hiện điều khiển Edit, dùng để gõ tên thư mục (nếu cần).

dir : là thư mục mặc định được chọn mỗi khi mở hộp thoại Select Folder.

Ví dụ :



23. FileGetTime("filename", [option], [format])

- Công dụng : trả về thông tin ngày và giờ của một tập tin.

- Xem bảng thông số hàm sau

filename	Đường dẫn tập tin cần lấy thông tin ngày và giờ.
option	**Tham số này quy định loại thông tin cần trả về** 0 = Thông tin ngày, giờ chỉnh sửa tập tin cuối cùng (mặc định). 1 = Thông tin ngày, giờ tạo tập tin. 2 = Thông tin ngày, giờ truy cập tập tin lần cuối cùng.
format	Tham số quy định kiểu kết quả trả về 0 (mặc định) = Thông tin ngày giờ sẽ trả về trong một mảng có sáu phần tử. Phần tử đầu tiên sẽ chứa thông tin năm (có bốn chữ số). Phần tử thứ hai chứa thông tin tháng (01 – 12). Phần tử thứ ba chứa ngày (01 – 31). Phần tử thứ tư chứa thông tin giờ (00 – 23). Phần tử thứ năm chứa phút (00 – 59). Phần tử thứ sau chứa thông tin giây (00 – 59). Nếu tham số format = 1, hàm sẽ trả về một chuỗi string có định dạng như sau YYYYMMDDHHMMSS (YYYY – năm, MM – tháng, DD – ngày, HH – giờ, MM – phút, SS – giây).

Ví dụ :
```
$t =  FileGetTime(@Windowsdir & "\Notepad.exe", 1)
    If Not @error Then
        $yyyymd = $t[0] & "/" & $t[1] & "/" & $t[2]
        MsgBox(0, "Ngay, thang, nam tao tap tin Notepad.exe", $yyyymd)
    EndIf
```

24. FileGetVersion("filename", ["stringname"])

- Công dụng : lấy thông tin phiên bản của một tập tin. Ngoài ra còn có thể lấy thêm một số thông tin khác.

- filename : là đường dẫn tên tập tin. stringname : là tên trường thông tin cần lấy về từ header trong thông tin tập tin. stringname có thể là : Comments, InternalName, ProductName, CompanyName, LegalCopyright, ProductVersion, FileDescription, LegalTrademarks, PrivateBuild, FileVersion, OriginalFilename, SpecialBuild.

Ví dụ : *$Prname = FileGetVersion("Explorer.exe", "ProductName")*

MsgBox(0, "Explorer version", $Prname)

$ver = FileGetVersion("Explorer.exe")

MsgBox(0, "Explorer version", $ver)

25. FileInstall("source", "dest", [flag])

- Công dụng : kết hợp một tập tin vào một đoạn mã lệnh script AutoIt. Khi đoạn mã lệnh được biên dịch, tập tin trong script AutoIt sẽ được đưa vào một ổ đĩa hoặc thư mục khác trên máy tính của bạn tùy thuộc vào tham số dest trong hàm.

- source : là đường dẫn đến tập tin cần đưa vào mã lệnh script AutoIt. dest là đường dẫn đích mà khi biên dịch, tập tin kết hợp sẽ được chuyển vào vị trí đó. flag là tham số tùy chọn, mặc định là 0. flag = 0, không ghi đè tập tin nếu có tập tin nào đó trùng tên ở thư mục đích. flag =1 tiến hành ghi đè tập tin nếu nó bị trùng ở đường dẫn đích.

Ví dụ : *$b = True*

If $b = True Then FileInstall("D:\rabbit.jpg", "C:\test.jpg")

Bây giờ bạn hãy tạo một tập tin có tên là rabbit.jpg đặt ở ổ D. Sau đó chép đoạn mã ví dụ trên vào cửa sổ SciTE, nhấn F5 để tiến hành chạy thử chương trình. Sau đó bạn qua ổ C để xem kết quả.

26. FileFindFirstFile("filename")

- Công dụng : tìm kiếm tập tin/thư mục theo yêu cầu của người dùng và trả về một "handle" chứa danh sách các tập tin/thư mục thỏa điều kiện. Hàm này kết hợp với hàm FileFindNextFile để liệt kê tập tin kết quả được tìm thấy.

- filename : là đường dẫn ổ đĩa hoặc thư mục cần tìm kiếm. Nếu không quy định đường dẫn ổ đĩa hoặc thư mục, hàm sẽ tìm kiếm trong thư mục chứa tập tin *.au3 (tập tin mã script AutoIt của bạn). Bạn có thể dùng các mẫu tự Wildcards như * hoặc ? để quy định tập tin trả về (* - đại diện cho chuỗi kí tự, ? - đại diện cho một kí tự).

- Hàm này nếu gọi thành công sẽ trả về "handle" để bạn dùng tiếp với hàm FileFindNextFile. Hàm trả về -1 nếu có lỗi xảy ra.

Ví dụ : *$search = FileFindFirstFile("D:*.*")*

If $search = -1 Then

MsgBox(0, "Error", "Khong co tap tin hoac thu muc nao thoa dieu kien tim kiem").

Exit

EndIf

27. FileFindNextFile(handle)

- Công dụng : trả về tập tin thỏa điều kiện tìm kiếm được gọi trong hàm FileFindFirstFile.

- handle : là biến chứa kết quả của hàm FileFindFirstFile.

- Nếu không có tập tin nào thỏa điều kiện tìm kiếm trong hàm FileFindFirstFile, thì @error sẽ gán bằng 1.

Ví dụ : *$search = FileFindFirstFile("D:*.*")*

```
If $search = -1 Then
   MsgBox(0, "Error", " Khong co tap tin hoac thu muc nao thoa dieu kien tim kiem ")
   Exit
EndIf
While 1
   $file = FileFindNextFile($search)
   If @error Then ExitLoop
   MsgBox(4096, "File:", $file)
Wend
```

28. IniDelete("filename", "section", ["key"])

- Công dụng : xóa một khóa key hoặc xóa một section trong một tập tin cấu hình ứng dụng ini.

- Cấu trúc tập tin *.ini như sau :

```
[SectionName]
Key=Value
```

Một ví dụ của tập tin *.ini là bạn hãy mở xem tập tin boot.ini có trong ổ C hoặc ổ nào đó chứa hệ điều hành Windows.

- Bảng thống số hàm

filename	Đường dẫn đến tập tin *.ini.
section	Tên section của tập tin ini.
key	Tham số tùy chọn. Tham số này là tên key cần xóa. Nếu tham số key không có, hàm sẽ xóa luôn section. Nếu bạn cung cấp tham số key là Default, hàm cũng sẽ xóa luôn section đó.

- Hàm này nếu gọi thành công sẽ trả về 1. Hàm trả về 0 nếu tập tin ini không tồn tại, hoặc tập tin có thuộc tính chỉ đọc không cho phép ghi.

Ví dụ : xóa section Configuration trong tập tin wincmd.ini (tập tin cấu hình trong chương trình quản lý tập tin Total Commander).

```
IniDelete("D:\Wincmd.ini", "Configuration", Default)
```

29. IniRead("filename", "section", "key", "default")

- Công dụng : đọc giá trị value của một khóa key trong một section nào đó của một tập tin INI.

- Bảng thông số hàm

filename	Là đường dẫn đến tập tin *.ini
section	Tên section của tập tin *.ini.
key	Tên khóa key trong tập tin *.ini cần đọc giá trị value.
default	Giá trị mặc định sẽ được trả về của hàm nếu khóa key yêu cầu cần đọc không được tìm thấy trong tập tin *.ini.

Ví dụ : minh họa trên tập tin cấu hình Wincmd.ini

30. IniReadSection("filename", "section")

- Công dụng : đọc các khóa key cùng với giá trị tương ứng của chúng trong một section.

- filename : là đường dẫn đến tập tin INI. Section là tên chương mà bạn muốn đọc key và value.

- Hàm này nếu gọi thành công sẽ trả về một mảng hai chiều. Trong đó mảng [0][0] sẽ chứa số kết quả trả về của khóa key. Các giá trị khóa key và value sẽ nằm tương ứng ở vị trí mảng [1][0] (key1), [1][1] (value1), [2][0] (key2), [2][1] (value2)… [n][0] (keyn), [n][1] (valuen).

Ví dụ : *$var = IniReadSection("D:\Wincmd.ini", "Configuration")*

```
If @error Then
        MsgBox(4096, "", "Da co loi xay ra trong qua trinh doc tap tin INI")
Else
        For $i = 1 To $var[0][0]
        MsgBox(0, "IniReadSection", "Key: " & $var[$i][0] & @CRLF & "Value: " & $var[$i][1])
        Next
EndIf
```

31. IniReadSectionNames("filename")

- Công dụng : đọc tất cả danh sách các section trong một tập tin INI.

- filename : là tên tập tin INI cần đọc.

- Hàm này nếu gọi thành công sẽ trả về một mảng. Trong đó phần tử mảng ở vị trí đầu tiên sẽ chứa số kết quả trả về. Các phần tử còn lại trong mảng chứa tên các section.

Ví dụ : xem danh sách các section trong tập tin boot.ini nằm trong ổ đĩa chứa hệ điều hành Windows.

```
$var = IniReadSectionNames("C:\boot.ini")
If @error Then
    MsgBox(4096, "", "Error occurred, probably no INI file.")
Else
    For $i = 1 To $var[0]
        MsgBox(0, "IniReadSectionNames", $var[$i])
    Next
EndIf
```

32. IniRenameSection("filename", "section", "newsection", [flag])

- Công dụng : sửa tên một section thành tên khác (newsection).

- Bảng thông số hàm

filename	Là đường dẫn đến tập tin *.ini
section	Tên section trong tập tin *.ini cần sửa.
new section	Tên của section mới.
flag	0 (mặc định) – Hàm này sẽ thất bại khi gọi nếu như có một section nào đó trong tập tin *.ini trùng tên với newsection. 1 – Trong trường hợp newsection có trùng với một section có sẵn trong tập tin *.ini, hàm sẽ xóa các khóa key trong section bị trùng đó (ghi đè section).

Ví dụ : *IniRenameSection("D:\Wincmd.ini", "left", "leftnew", 1)*

33. IniWrite("filename", "section", "key", "value")

- Công dụng : ghi một giá trị value đến một khóa key (bạn có thể dùng hàm này để sửa một giá trị trong một khóa key).

- Bảng thông số hàm

filename	Đường dẫn đến tập tin *.ini
section	Tên của section.
key	Tên khóa key cần ghi giá trị.

value	Giá trị để ghi hoặc sửa.

Ví dụ : *IniWrite("D:\Wincmd.ini", "Layout", "ButtonBar", "0")*

34. IniWriteSection("filename", "section", "data", [index])

- Công dụng : ghi dữ liệu cặp đôi key và value đến một section trong một tập tin INI.

- Bảng thông số hàm

filename	Đường dẫn đến tập tin *.ini cần ghi.
section	Tên section cần ghi dữ liệu key và value.
data	Dữ liệu key và value cần ghi đến section. Dữ liệu data có thể là một chuỗi string hoặc là một mảng. Nếu bạn thích dữ liệu tham số data là một chuỗi, bạn cần phải đặt theo quy định sau : $sData = "Key1=Value1" & @LF & "Key2=Value2" & @LF & "Key3=Value3" & @LF & "Keyn = Valuen" Nếu bạn cung cấp tham số data là một mảng, bạn cần phải cung cấp một mảng hai chiều. Trong đó chiều thứ nhất có độ dài là số cặp đôi key và value mà bạn cần ghi. Trong đó chiều thứ hai luôn là hai. Bạn nên đặt biến mảng cần ghi theo quy định sau : $Data[n][2] = [["Key1", "Value1"], ["Key2", "Value2"], ["Key3", "Value3"], ["Keyn", "Valuen"]]
index	Tham số tùy chọn. Nếu bạn quy định data là một mảng. Bạn có thể quy định cho cặp khóa key và value bắt đầu cần ghi. Ví dụ trong khai báo $Data[n][2] = [["Key1", "Value1"], ["Key2", "Value2"], ["Key3", "Value3"], ["Keyn", "Valuen"]] ở trên. Nếu bạn muốn ghi bắt đầu từ khóa key2 – value2, bạn quy định index = 1. Tương tự nếu bạn muốn ghi bắt đầu tử khóa key3 – value3, bạn quy định index = 2.

Ví dụ :

$Data1 = "Name=Mr. Trinh" & @LF & "Class=DH10TH" & @LF & "MSSV=DTH092088" & @LF & "Age = 20"

Dim $Data2[4][2] = [["Name", "DTTL"],["Class", "DH10TH"],["MSSV", "DTH0920XX"],["Age", "20"]]

IniWriteSection("D:\Wincmd.ini", "LeftHistory", $Data1)

IniWriteSection("D:\Wincmd.ini", "Configuration", $Data2, 0)

35. FileChangeDir("path")

- Công dụng : thay đổi đường dẫn thư mục hiện hành đến thư mục khác.

- path : là đường dẫn mà bạn muốn chuyển đổi tới làm thư mục hiện hành.

Ví dụ : *FileChangeDir(@MyDocumentsDir)*; thư mục My Documents.

36. DriveStatus("path")

- Công dụng : lấy thông tin trạng thái của một phân vùng hay một loại thiết bị media nào đó trên máy tính của bạn.

- path : là đường dẫn đến phân vùng hoặc thiết bị media mà bạn cần lấy thông tin. Hàm này gọi thanh công sẽ trả về một chuỗi chứa các thông tin trạng thái, bao gồm :

Giá trị trả về	Mô tả
UNKNOWN	Ổ đĩa hoặc thiết bị có thể không được định dạng.
READY	Các loại phân vùng của ổ đĩa cứng hoặc các thiết bị lưu trữ gắn rời.
NOTREADY	Các ổ đĩa mềm hoặc ổ đĩa quang nhưng không có đĩa trong ổ.
INVALID	Kí tự ổ đĩa bạn cung cấp không tồn tại hoặc những ổ đĩa ánh xạ kết nối mạng không cho truy cập.

Ví dụ : *MsgBox(0, "Drive Get Status", DriveStatus("F:\"))*;

37. FileCreateShortcut("file", "lnk", ["workdir"], ["args"], ["desc"], ["icon"], ["hotkey"], [icon number], [state])

- Công dụng : tạo một shortcut cho một tập tin.

- Bảng thông số hàm

file	Đường dẫn đầy đủ đến tập tin cần tạo shortcut.
lnk	Đường dẫn đầy đủ của thư mục chứa shorcut.

workdir	Thư mục làm việc của tập tin được tạo shortcut (cung cấp đường dẫn thư mục đến tập tin cần tạp shortcut).
args	Thêm các đối số cho shortcut được tạo.
desc	Mô tả cho tập tin shortcut.
icon	Đường dẫn đầy đủ đến tập tin thư viện icon (*.dll) cần cùng cho biểu tượng shortcut.
hotkey	Phím nóng để gọi shortcut.
icon number	Chỉ mục của icon trong tập tin thư viện icon (*.dll) cần dùng.
state	Trạng thái của cửa sổ chương trình xuất hiện khi người dùng nhấp đôi vào shortcut. @SW_SHOWNORMAL (hiện cửa sổ thông thường cho người dùng), @SW_SHOWMINNOACTIVE (hiện cửa sổ thu nhỏ trên thanh taskbar) hoặc @SW_SHOWMAXIMIZED (phóng lớn toàn màn hình cửa sổ chương trình).

Ví dụ : *FileCreateShortcut(@WindowsDir & "\Explorer.exe",@DesktopDir & "\Shortcut Test.lnk",@WindowsDir,"/e,c:\", "This is an Explorer link;-)", @SystemDir & "\shell32.dll", "^!t", "4", @SW_MINIMIZE)*; tạo shortcut cho chương trình Window Explorer và đặt nó trên màn hình Desktop với phím nóng là Ctrl – Alt – T ("^!t" - bạn có thể xem thêm hàm Send() được giới thiệu trong những bài viết ở các kì tới để biết thêm cách tạo phím nóng khác.)

Bạn thay tham số icon number = 4 trong chương trình trên thành các số khác nhau để có được các biểu tượng khác nhau khi tạo shortcut trên màn hình Desktop.

38. FileOpen("filiename", [mode])

- Công dụng : mở một tập tin văn bản để đọc hoặc ghi hoặc cả hai.

- filename : là đường dẫn đến tập tin cần đọc và ghi. Mode là tham số quy định để mở tập tin. Nó có các giá trị sau :

0 = chế độ chỉ đọc (mặc định).

1 = chế độ ghi (ghi vào cuối tập tin).

2 = chế độ ghi nhưng xóa nội dung tập tin trước khi ghi.

8 = Tạo cấu trúc thư mục nếu như đường dẫn tập tin do người dùng cung cấp không tồn tại.

16 = dùng chế độ thao tác cho tập tin nhị phân.

32 = Dùng Unicode UTF16 Little Endiancho thao tác đọc và ghi.

64 = Dùng Unicode UTF16 Big Endiancho thao tác đọc và ghi.

128 = Dùng Unicode UTF8 (with BOM) cho thao tác đọc và ghi.

256 = Dùng chế độ Unicode UTF8 (without BOM) cho thao tác đọc và ghi.

- Hàm này nếu gọi thành công sẽ trả về một "handle" để dùng kết hợp với một số hàm thao tác đọc ghi tập tin văn bản khác.

Ví dụ : *$file = FileOpen("D:\test.txt", 10)*; (tham số mode = 10, kết hợp hai mode 2 và 8).

39. FileRead("filehandle/filename", [count])

- Công dụng : đọc nội dung của một tập tin văn bản.

- filehandle : là biến "handle" được trả về từ hàm FileOpen. Nếu bạn không có biến "handle", bạn có thể dùng một đường dẫn đầy đủ đên tập tin văn bản cần đọc. count là số kí tự mà bạn cần đọc từ tập tin văn bản. Nếu bạn không cung cấp tham số này thì toàn bộ nội dung của một tập tin văn bản sẽ được đọc.

Ví dụ :

Bạn có thể xóa dòng chứa $handle sau đó gõ đoạn mã *MsgBox(0, "File Read", FileRead("D:\StudentInfo.txt"))* cũng có kết quả tương tự.

40. FileReadLine("filehandle/filename", [line])

- Công dụng : đọc một dòng nào đó của tập tin văn bản.

- filehandle : là biến "handle" được trả về từ hàm FileOpen. Nếu bạn không có biến "handle", bạn có thể dùng một đường dẫn đầy đủ đến tập tin văn bản cần đọc. line là số thứ tự dòng mà bạn muốn đọc, dòng đầu tiên có giá trị là 1.

Ví dụ : *$handle = FileOpen("D:\StudentInfo.txt", 0)*

MsgBox(0, "File Read", FileReadLine($handle, 2)); trả về "Lớp : DH10SM".

41. FileWrite("filename/filehandle", "text")

- Công dụng : ghi nội dung văn bản đến phần cuối của tập tin văn bản đã mở cho phép ghi.

- filehandle : là biến "handle" được trả về từ hàm FileOpen. Nếu bạn không có biến "handle", bạn có thể dùng một đường dẫn đầy đủ đến tập tin văn bản cần ghi. text là

văn bản cần ghi.

Ví dụ : *$file = FileOpen("D:\StudentInfo.txt", 1); mở tập tin để ghi.*

```
If $file = -1 Then
        MsgBox(0, "Error", "Khong the mo tap tin")
        Exit
EndIf
FileWrite($file, @CRLF&"Name : Ho Thi Thanh Thien"&@CRLF&"Lop : DH11TH")
```

42. FileWriteLine("filename/filehandle", "dataline")

- Công dụng : ghi văn bản đến phần cuối của tập tin văn bản nhưng ghi theo dòng.

- filehandle : là biến "handle" được trả về từ hàm FileOpen. Nếu bạn không có biến "handle", bạn có thể dùng một đường dẫn đầy đủ đến tập tin văn bản cần ghi. dataline là văn bản cần ghi.

Ví dụ : *$file = FileOpen("D:\StudentInfo.txt", 1)*

```
If $file = -1 Then
   MsgBox(0, "Error", "Khong the mo tap tin")
   Exit
EndIf
FileWriteLine($file, "Viet Nam dat nuoc que huong chung toi")
FileWriteLine($file, "Co dong dua xanh xa khuat chan troi")
```

43. FileClose(filehandle)

- Công dụng : đóng tập tin trước đó được mở bằng hàm FileOpen.

- filehanle là biến "handle" được dùng trong quá trình sử dụng hàm FileOpen.

- Bạn nên dùng hàm này sau khi xử lý xong tập tin văn bản bằng các lệnh như FileWrite, FileRead,…

Ví dụ : *$handle = FileOpen("D:\StudentInfo.txt", 0)*

```
MsgBox(0, "FileClose", FileRead($handle, 6))
FileClose($handle)
```

44. FileFlush(handle)

- Công dụng : chuyển vùng nhớ của bộ đệm tập tin đến đĩa.

- handle là biến "handle" được dùng trong quá trình sử dụng hàm FileOpen.

Ví dụ : *$handle = FileOpen("D:\StudentInfo.txt", 0)*

```
FileFlush($handle)
FileClose($handle)
```

Trong quá trình sử dụng hàm xử lý tập tin thư mục, nếu bạn thường dùng các thư mục hệ thống của hệ điều hành trong quá trình xử lý, bạn thường hay gõ đường dẫn đầy đủ của thư mục đó. Nhưng AutoIt có cung cấp một số macro thư mục cho người dùng để

tiện cho người dùng xử lý tập tin hoặc thư mục mà không cần gõ đường dẫn đầy đủ của thư mục hoặc tập tin đó. Dưới đây là danh sách các macro thư mục : (trong máy tính của tôi, hệ điều hành chính cài ở ổ C – hệ điều hành Windows XP).

Macro	Mô tả
Macro thư mục áp dụng cho tất cả người dùng.	
@AppDataCommonDir	C:\Documents and Settings\All Users\Application Data
@DesktopCommonDir	C:\Documents and Settings\All Users\Desktop
@DocumentsCommonDir	C:\Documents and Settings\All Users\Documents
@FavoritesCommonDir	C:\Documents and Settings\All Users\Favorites
@ProgramsCommonDir	C:\Documents and Settings\All Users\Start Menu\Programs
@StartMenuCommonDir	C:\Documents and Settings\All Users\Start Menu
@StartupCommonDir	C:\Documents and Settings\All Users\Start Menu\Programs\ Startup
Macro thư mục áp dụng cho tài khoản người dùng hiện tại (tài khoản hiện tại của tôi là Mr. Trinh).	
@AppDataDir	C:\Documents and Settings\Mr. Trinh\Application Data
@DesktopDir	C:\Documents and Settings\Mr. Trinh\Desktop
@MyDocumentsDir	C:\Documents and Settings\Mr. Trinh\My Documents
@FavoritesDir	C:\Documents and Settings\Mr. Trinh\Favorites
@ProgramsDir	C:\Documents and Settings\Mr. Trinh\Start Menu\Programs
@StartMenuDir	C:\Documents and Settings\Mr. Trinh\Start Menu
@StartupDir	C:\Documents and Settings\Mr. Trinh\Start Menu\Programs\ Startup
@UserProfileDir	C:\Doucments and Settings\Mr. Trinh
Macro thư mục khác	
@ProgramFilesDir	C:\Program Files

@CommonFilesDir	C:\Program Files\Common Files
@WindowsDir	C:\Windows
@SystemDir	C:\Windows\System32
@TempDir	C:\Documents and Settings\Mr. Trinh\Local Settings\Temp
@WorkingDir	Thư mục chứa tập tin *.au3 mà bạn đang lập trình.

Hàm tiện ích âm thanh

1. Beep([fre], [duration])

- Công dụng : phát tiếng beep.

- fre : tần số phát âm thanh, có thể có giá trị từ 37 đến 32767Hz, giá trị mặc định là 500Hz. duration : chiều dài của tiếng beep (đơn vị : ms), mặc định là 1000 ms.

Ví dụ : *Beep(600, 6000)*

2. SoundPlay("filename", [wait])

- Công dụng : phát một tập tin âm thanh.

- filename : đường dẫn đến tập tin cần chơi. Hai dạng tập tin được hỗ trợ là .mp3 và .wav. wait là tham số tùy chọn, wait = 1 đoạn mã lệnh lập trình sau câu lệnh SoundPlay sẽ được tạm dừng trong quá trình chơi nhạc, nó sẽ bắt đầu thực hiện lại đoạn mã lệnh sau khi đoạn nhạc kết thúc. wait = 0, đoạn mã lệnh lập trình sau câu lệnh SoundPlay vẫn được thực thi trong quá trình chơi file.

Ví dụ : *SoundPlay("D:\USB\Music\Cay dan bo quen.mp3", 1)*

3. SoundSetWaveVolume(percent)

- Công dụng : thiết đặt âm lượng wave của hệ thống.

- percent là tỉ lệ phần trăm muốn thiết đặt âm lượng, có giá trị tử 0 đến 100.

Ví dụ : *SoundSetWaveVolume(80)*

4. PixelChecksum(left, top, right, bottom, [step], [hwnd])

- Công dụng : tính số điểm ảnh trên một khu vực nào đó của màn hình (khu vực này có dạng hình chữ nhật).

- left : tọa độ bên trái của hình chữ nhật (X1). right : tọa độ bên phải của hình chữ nhật (X2). top : tọa độ cạnh trên của hình chữ nhật (Y1). bottom : tọa độ cạnh dưới của hình chữ nhật (Y2). step : thay vì kiểm tra tuần tự từng pixel, dùng một con số lớn hơn 1 để bỏ qua pixel kiểm tra. Mặc định là 1, không khuyến khích bạn dùng một giá trị lớn hơn 1. hwnd là tham số biến "handle" được dùng để kiểm tra số pixel của một cửa sổ nào đó. Bạn có thể xem tham số handle trong chương trình AutoIt Windows Info.

5. PixelGetColor(x, y, [hwnd])

- Công dụng : trả về giá trị màu tại một điểm ảnh nào đó.

- x, y tương ứng với tọa độ X, Y trên màn hình cần kiểm tra. hwnd là biến handle tới cửa sổ chương trình cần kiểm tra. Nếu không sẽ sử dụng cửa sổ chương trình hiện tại.

- Hàm này nếu gọi thành công sẽ trả về số thập phân tương ứng với giá trị màu.

Ví dụ : *$var = PixelGetColor(0, 0)*

MsgBox(0,"Gia tri mau dang thap phan ", $var)

MsgBox(0,"Gia tri mau dang hexa ", Hex($var, 6)); chuyển sang dạng số hexadecimal.

Hàm xử lý bàn phím

1. Send("key", [flag])

- Công dụng : giả lập phím nhấn đến cửa sổ đang kích hoạt.

- key : là phím bạn cần giả lập. Ví dụ như {ENTER} (nhấn phím ENTER). flag là tham số tùy chọn, mặc định là 0. Trường hợp flag = 0, nếu như tham số key trong hàm có chứa các kí tự như + và ! thì thay vì gửi phím + và ! nó sẽ gửi phím Shift và Alt. Trường hợp flag = 1, nếu trong tham số key của hàm có chứa các kí tự như + và ! thì chỉ gửi như nguyên gốc của nó đó là + hoặc !. Bạn có thể tham khảo các quy định xâu chuỗi tương ứng với các phím được nhấn như sau.

Tham số key (trường hợp cờ flag = 0	Phím trên bàn phím sẽ được nhấn
{!}	! (nếu flag = 1, phím ALT được nhấn)
{#}	#
{+}	+ (nếu flag = 1, phím SHIFT được nhấn)

{^}	^
{{}	{
{}}	}
{SPACE}	SPACE
{ENTER}	ENTER
{ALT}	ALT
{BACKSPACE} hoặc {BS}	BACKSPACE
{DELETE} hoặc {DEL}	DELETE
{UP}	↑
{DOWN}	↓
{LEFT}	←
{RIGHT}	→
{HOME}	HOME
{END}	END
{ESCAPE} hoặc {ESC}	ESC
{INSERT} or {INS}	INSERT
{PGUP}	PageUp
{PGDN}	PageDown
{F1} - {F12}	Các phím chức năng từ F1 đến F12
{TAB}	TAB
{PRINTSCREEN}	Phím Print Screen
{LWIN}	Phím Windows nằm bên trái bàn phím

{RWIN}	Phím Windows nằm bên phải bàn phím
{NUMLOCK on}	Bật phím Num Lock
{NUMLOCK off}	Tắt phím Num Lock
{NUMLOCK toggle}	Chuyển trạng thái đèn Num Lock tử on sang off hoặc ngược lại.
{CAPSLOCK off}	Tắt đèn Caps Lock, bạn có thể sử dụng tham số on hoặc toggle tương tự như Num Lock.
{SCROLLLOCK toggle}	Chuyển trạng thái đèn từ on sang off hoặc ngược lại. Bạn có thể sử dụng tham số on hoặc off tương tự như Num Lock.
{BREAK}	Ctrl+Break
{PAUSE}	PAUSE
{NUMPAD0} - {NUMPAD9}	Cụm phím số bên phải bàn phím từ 0 đến 9
{NUMPADMULT}	*
{NUMPADADD}	+
{NUMPADSUB}	-
{NUMPADDIV}	/
{NUMPADENTER}	Phím Enter trong cụm phím Numpad
{APPSKEY}	Tương tự như khi bạn nhấn chuột phải vào một vị trí nào đó trên màn hình
{LALT}	Phím ALT bên trái bàn phím
{RALT}	Phím ALT bên phải bàn phím
{LCTRL}	Phím Ctrl bên trái bàn phím
{RCTRL}	Phím Ctrl bên phải bàn phím
{LSHIFT}	Phím Shift bên trái bàn phím

{RSHIFT}	Phím Shift bên phải bàn phím
{SLEEP}	SLEEP
{ALTDOWN}	Nhấn chìm phím Alt cho đến khi {ALTUP} được gọi
{SHIFTDOWN}	Nhấn chìm phím Shift cho đến khi {SHIFTUP} được gọi
{CTRLDOWN}	Nhấn chìm phím Ctrl cho đến khi {CTRLUP} được gọi
{LWINDOWN}	Nhấn chìm phím Windows (nằm bên trái bàn phím cho đến khi {LWINUP} được gọi
{RWINDOWN}	Nhấn chìm phím Windows (nằm bên phải bàn phím cho đến khi {RWINUP} được gọi
{ASC nnnn}	Phím Alt + với một phím nnnn nào đó.
{BROWSER_BACK}	Nút Back trong trình duyệt Web (dành cho XP/2000)
{BROWSER_FORWARD}	Nút Forward trong trình duyệt Web (dành cho XP/2000)
{BROWSER_REFRESH}	Nút Refresh trong trình duyệt Web (dành cho XP/2000)
{BROWSER_STOP}	Nút Stop trong trình duyệt Web (dành cho XP/2000)
{BROWSER_SEARCH}	Nút Search trong trình duyệt Web (dành cho XP/2000)
{BROWSER_FAVORITES}	Nút Favorites trong trình duyệt Web (dành cho XP/2000)
{BROWSER_HOME}	Mở trình duyệt web mặc định và mở trang chủ
{VOLUME_MUTE}	Tắt âm thanh (dành cho XP/2000)
{VOLUME_DOWN}	Giảm âm lượng âm thanh (dành cho XP/2000)
{VOLUME_UP}	Tăng âm lượng âm thanh (dành cho XP/2000)

{MEDIA_NEXT}	Chọn file kế tiếp (dành cho XP/2000)
{MEDIA_PREV}	Chọn file trước đó. (dành cho XP/2000)
{MEDIA_STOP}	Dừng phát file. (dành cho XP/2000)
{MEDIA_PLAY_PAUSE}	Chơi hoặc tạm dừng file. (dành cho XP/2000)
{LAUNCH_MAIL}	Khởi động trình duyệt mail mặc định trên máy tính của bạn. (dành cho XP/2000)
{LAUNCH_MEDIA}	Khởi động trình Media Player mặc định trên máy tính của bạn. (dành cho XP/2000)

- Để gửi một phím chữ thay vì gọi lệnh Send("{A}"), bạn có thể dùng ASC + với mã ASCII của phím cần nhấn. Ví dụ như bạn muốn nhấn chữ A, bạn hãy gõ lệnh Send("{ASC 065}") (65 là mã ASCII của A). Để gửi một kí tự ở ngôn ngữ khác, bạn có thể cần phải biết mã decimal hoặc hex của kí tự đó.

- Nếu bạn cần nhấn phím một số lần nào đó. Ví dụ cần nhấn phím A bốn lần, bạn có thể gõ Send("{A 4}"). Nếu bạn thấy bất tiện, bạn có thể dùng vòng lặp để lặp lại thao tác giả lập nhấn phím.

- Nếu bạn muốn nhấn chìm phím nào đó. Ví dụ cần nhấn chìm phím a, bạn có thể gõ hàm Send("{a down}"). Còn để nhả phím ra, bạn gõ Send("{a up}").

2. HotKeySet("key", ["function"])

- Gán phím nóng cho một hàm trong chương trình. Khi bạn nhấn phím nóng này thì function sẽ được thực thi.

- key là phím nóng mà bạn cần gán, nhưng bạn không được dùng các phím nóng như Ctrl+Alt+Del, F12, Windows+B,D,E,F,L,M,R,U và Win+Shift+M, Alt, Ctrl, Shift, Win, hoặc các phím được quy định bởi các chương trình phần mềm khác. function là tên hàm mà bạn cần gọi (hàm này bạn định nghĩa trong chương trình).

Ví dụ :
```
Global $Paused
HotKeySet("{PAUSE}", "TogglePause")
HotKeySet("{ESC}", "Terminate")
HotKeySet("+d", "ShowMessage")  ;Shift-D
While 1
    Sleep(100)
WEnd
Func TogglePause()
    $Paused = NOT $Paused
    While $Paused
```

sleep(100)

ToolTip('Script is "Paused"',0,0); Hàm ToolTip có công dụng xuất hiện một mẫu ghi chú nhỏ màu vàng, giống popup.

WEnd

ToolTip("")

EndFunc

Func Terminate()

Exit 0; thoát khỏi chương trình

EndFunc

Func ShowMessage()

MsgBox(4096,"","This is a message.")

EndFunc

3. SendKeepActive("Title")

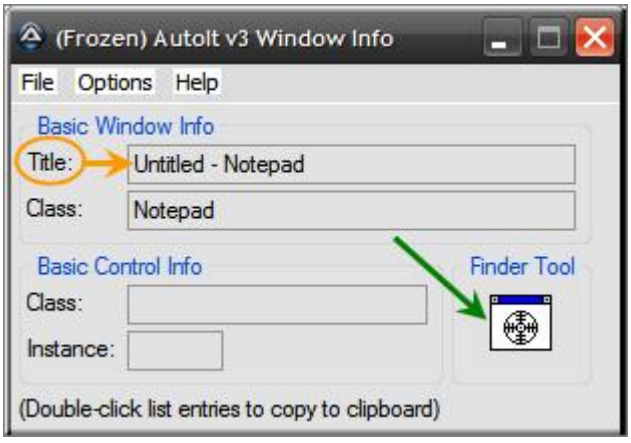
- Công dụng : giữ một cửa sổ chương trình nào đó luôn luôn được kích hoạt trong khi gọi hàm Send.

- title : là tên tiêu đề của cửa sổ. Bạn có thể thấy tên tiêu đề của một cửa sổ bất kì bằng cách vào Start > All Programs > AutoIt v3 > AutoIt Window Info. Cửa sổ chương trình hiện ra, bạn có thể nhấn giữ trái chuột vào công cụ Finder Tool, sau đó rê vào cửa sổ cần lấy thông tin Title (kết quả sẽ hiển thị trong trường Title nhóm Basic Window Info).

Ví dụ : *SendKeepActive("Untitled - Notepad")*

Sleep(2000); tạm dừng chương trình trong 2s.

Send("{B 20}")



Hàm toán học

1. Abs(expression)

- Công dụng : tính trị tuyệt đối của một con số.

- expression : là con số cần lấy giá trị tuyệt đối. Nếu expression là một chuỗi, hàm trả về 0.

2. ACos(expression)

- Công dụng : tính arcCosine của một con số. (hàm trả về theo đơn vị radians)
- expression : là con số cần tính arcCosine. Giá trị của nó phải nằm từ -1 đến 1.

3. ASin(expression)

- Công dụng : tính arcsine của một con số. (hàm trả về theo đơn vị radians)
- expression : là con số cần tính arcsine. Giá trị của nó phải nằm từ -1 đến 1.

4. Atan(expression)

- Công dụng : tính arctangent của một số. (hàm trả về theo đơn vị radians)
- expression : là một con số thực nào đó bất kì.

5. Sin(expression), Cos(expression), Tan(expression)

- Công dụng : tính sin, cos, tan của một số.
- expression : là con số cần tính. Expression phải là con số radians.

6. Ceiling(expression)

- Công dụng : làm tròn một con số đến con số nguyên kế tiếp lớn hơn nó.
- expression : là con số cần làm tròn.

Ví dụ : *$msg = ""*

$msg = $msg & "Ceiling(4.1247899) = " & Ceiling(4.1247899) & @CR; Ceiling(4.1247899) = 5

$msg = $msg & "Ceiling(4.5) = " & Ceiling(4.5) & @CR; Ceiling(4.5) = 5

$msg = $msg & "Ceiling(4.3) = " & Ceiling(4.3) & @CR; Ceiling(4.3) = 5

$msg = $msg & "Ceiling(4) = " & Ceiling(4) & @CR; Ceiling(4) = 4

$msg = $msg & "Ceiling(-4.3) = " & Ceiling(-4.3) & @CR; Ceiling(-4.3) = -4

$msg = $msg & "Ceiling(-4.5) = " & Ceiling(-4.5) & @CR; Ceiling(-4.5) = -4

$msg = $msg & "Ceiling(-4.8) = " & Ceiling(-4.8) & @CR; Ceiling(-4.8) = -4

$msg = $msg & "Ceiling(-4) = " & Ceiling(-4) & @CR; Ceiling(-4) = -4.

MsgBox(64, "Testing", $msg)

7. Floor(expression)

- Công dụng : làm tròn xuống một con số đến số nguyên nhỏ hơn nó.
- expression : là con số cần làm tròn.

Ví dụ : *$msg = ""*

$msg = $msg & "Floor(4.1247899) = " & Floor(4.1247899) & @CR; Floor(4.1247899) = 4

$msg = $msg & "Floor(4.5) = " & Floor(4.5) & @CR; Floor(4.5)= 4

$msg = $msg & "Floor(4.3) = " & Floor(4.3) & @CR; Floor(4.3) = 4

$msg = $msg & "Floor(4) = " & Floor(4) & @CR; Floor(4) = 4

$msg = $msg & "Floor(-4.3) = " & Floor(-4.3) & @CR; Floor(-4.3) = -5

$msg = $msg & "Floor(-4.5) = " & Floor(-4.5) & @CR; Floor(-4.5) = -5

$msg = $msg & "Floor(-4.8) = " & Floor(-4.8) & @CR; Floor(-4.8) = -5

$msg = $msg & "Floor(-4) = " & Floor(-4) & @CR; Floor(-4) = -4

MsgBox(64, "Testing", $msg)

8. Round(expression, [decimal])

- Công dụng : Làm tròn một con số.

- expression : là con số cần làm tròn. decimal là tham số tùy chọn vị trị cần làm tròn (-1 : hàng chục, -2 : hàng trăm, 0 : hàng đơn vị, 1: chữ số lẻ thử nhất, 2: chữ số lẻ thứ hai). Hàm này giống như hàm Round trong Microsoft Excel. Nếu tham số này không có, hàm sẽ làm tròn số đến số nguyên.

Ví dụ : *$x = Round(-1.582, 1) ;* trả về -1.6

$y = Round(3.1415, 9) ; không thay đổi

$z = Round(123.578) ; trả về 124

MsgBox(0, "Round", $x & " " & $y & " " & $z)

9. Mod(value1, value2)

- Công dụng : lấy phần dư của phép chia value1 cho value2.

- value1: số bị chia, value2: số chia.

10. Random([min], [max], [flag])

- Công dụng : tạo một số ngẫu nhiên trong đoạn [min;max].

- min : giá trị nhỏ nhất tạo ngẫu nhiên, nếu không gán, mặc định là 0. max : giá trị lớn nhất tạo ngẫu nhiên, nếu không gán, mặc định là 1. flag : tham số tùy chọn, nếu flag = 1 hàm sẽ trả về một con số ngẫu nhiên có giá trị nguyên. Nếu flag không bằng 1 thì hàm sẽ trả về một con số thực (mặc định của hàm này)

11. Sqrt(expression)

- Công dụng : tính căn bậc hai của một con số.

- expression : con số cần tính căn bậc hai. Nếu nó âm hàm sẽ trả về giá trị -1.

12. Exp (expression)

- Công dụng : tính e^expression.

- e : là con số được cho giá trị khoảng 2.71828182845905.

13. Log(expression)

- Công dụng : trả lại giá trị hàm loga cơ số tự nhiên ln(expression).

Ví dụ : *Log(1000);* trả về 6.90775527898214

14. BitAND(value1, value2, [value_n])

- Công dụng : thực hiện phép AND theo từng bit. Trên mỗi cặp bit tương ứng nhau của hai toán hạng, nếu cả hai bit đều là 1 thì sẽ trả về 1, ngược lại thì 0.

- value1, value2 : là con số mà bạn cần thực hiện phép AND, bạn có thể thực hiện tối đa

255 số cùng một lúc.

Ví dụ : *$x = BitAND(12, 7)*; trả về 4 bởi vì 12 (10) = 1100 (2), 7 (10) = 0111 (2). Như vậy 1100 AND 0111 = 0100.

14. BitOR(value1, value2, [value_n])

- Công dụng : thực hiện phép OR theo từng bit. Trên mỗi cặp bit tương ứng nhau của hai toán hạng, nếu một trong hai bit là 1 thì sẽ trả về 1, ngược lại thì trả về 0 (hai bit đều là 0).

- value1, value2 : là con số mà bạn cần thực hiện phép OR, bạn có thể thực hiện tối đa 255 số cùng một lúc.

Ví dụ : *$x = BitOR(12, 6)*; trả về 14 bởi vì 12 (10) = 1100 (2), 6 (10) = 0110 (2). Như vậy 1100 AND 0110 = 1110.

15. BitNOT(value)

- Công dụng : lấy phần bù của từng bit (tức đảo 0 thành 1 và ngược lại).

- value : là con số mà bạn cần thực hiện phép đảo bit. Tuy nhiên giá trị của value sẽ được chuyển đổi thành con số nhị phân dạng 32-bit.

Ví dụ : *$x = BitNOT(5)*; trả về -6. Bởi vì

5 = 00000000000000000000000000000101(2).

BitNOT(5) = 11111111111111111111111111111010(2) = -2^31 + 2^30 + 2^29+….2^3+2^1 = -6.

14. BitXOR(value1, value2, [value_n])

- Công dụng : thực hiện phép XOR theo từng bit. Phép XOR trả về theo từng cặp bit là 1 nếu một trong hai bit là 1 và bit còn lại là 0, ngược lại trả về 0.

- value1, value2 : là con số mà bạn cần thực hiện phép XOR, bạn có thể thực hiện tối đa 255 số cùng một lúc.

Ví dụ : *$x = BitXOR(12, 10)*; trả về 6 bởi vì 12 (10) = 1100 (2), 10 (10) = 1010 (2). Như vậy 1100 XOR 1010 = 0110.

15. SRandom(Seed)

- Công dụng : tạo con số mồi cho hàm Random.

- Seed : là con số mồi, nó có thể có giá trị từ -2^31 đến 2^31-1.

- Thông tin thêm : các hàm tạo số ngẫu nhiên trong các ngôn ngữ lập trình được phát sinh từ một hàm toán học như hàm đồng dư hay Fibonacci hoặc một thuật giải tạo số ngẫu nhiên khác. Tuy nhiên để tạo số ngẫu nhiên cho các hàm này, người ta cần tạo con số mồi. Sau đó các số ngẫu nhiên được sinh ra sau đó sẽ dựa trên các con số mồi đó. Giá trị của con số mồi được các hàm tạo số ngẫu nhiên trong các ngôn ngữ lập trình được tính dựa vào ngày giờ của đồng hồ máy tính hoặc mã quốc gia và múi giờ để phát sinh con số mồi này. Bây giờ, bạn hãy xem ví dụ về bộ sinh số ngẫu nhiên đồng dư như sau :

Bộ sinh đồng dư có dạng : x(i) = [a*x(i-1) + b] mod M. Trong đó a và b là hai con số nguyên cho trước, số nguyên M là số xấp xỉ hoặc bằng với số nguyên lớn nhất trong máy tính. Bạn để ý nếu ta muốn có x(1) thì ta phải có con số x(0), muốn có x(2) thì phải có x(1), muốn có x(k) phải có x(k-1). Vì vậy ta muốn sinh dãy các số ngẫu nhiên, thì ta

phải cung cấp con số x(0) (con số mồi), rồi từ đó ta mới các con số x(1), x(2), x(3),…

Ví dụ : *SRandom(9)*

MsgBox(0, "Binary", Random(0, 9, 1)); bạn hãy chạy chương trình này trong hai trường hợp có hàm SRandom và không có hàm SRandom để thấy rõ hơn công dụng của hàm.

- Khi bạn chạy ví dụ trên, bạn sẽ thấy mỗi lần chạy hàm sẽ luôn tạo một con số không thay đổi, đó là vì bạn tạo con số mồi rồi nên giá trị của nó sẽ cố định. Tuy vậy, nếu bạn thêm một dòng lệnh tạo số ngẫu nhiên trong ví dụ trên thì sẽ tạo số ngẫu nhiên khác.

16. BitShift(value, shift)

- Công dụng : thực hiện thao tác dịch trái hoặc dịch phải số nhị phân.

- value : là con số cần dịch chuyển (dạng thập phân) – value sẽ được chuyển đổi sang số nhị phân dạng 32-bit trước khi dịch. shift là số bit cần dịch chuyển, mặc định hàm sẽ tiến hành dịch phải, nếu bạn cho tham số shift là số âm hàm sẽ tiến hành dịch trái.

- Phép dịch trái là dịch chuyển tất cả các bit sang trái bởi số vị trí được quy định trong tham số shift. Các vị trí bit rỗng bên phải của số nhị phân của thao tác này sẽ được điền đầy bởi 0.

- Phép dịch phải là phép dịch tất cả các bit sang phải bởi số vị trí được quy định trong tham số shift. Các vị trí bit được dịch sang bên phải của thao tác này đều bị loại bỏ. Số bit bên trái sẽ được điền đầy bởi 0 nếu hầu hết các bit quan trọng (bit xa nhất theo hướng trái) của value (dạng nhị phân) là 0, và được điền đầy bởi 1 nếu các bit quan trọng là 1.

Ví dụ : *MsgBox(0, "BitShift", BitShift(1, -10))*; = 1024 bởi vì

1(10) = 0000 0000 0000 0000 0000 0000 0000 0001 (2) -> kết quả dịch trái 10 bit sẽ là :

0000 0000 0000 0000 0000 0100 0000 0000 (2)

MsgBox(0, "BitShift", BitShift(65535, 8)); = 255 bởi vì

65535 (10) = 0000 0000 0000 0000 1111 1111 1111 1111 (2) -> kết quả dịch phải 8 bit sẽ là :

0000 0000 0000 0000 0000 0000 1111 1111 (2)

17. BitRotate(value, shift, [size])

- Công dụng : thực hiện thao tác quay trái hoặc quay phải số nhị phân.

- value : là con số cần thực hiện phép quay. shift là số lần thực hiện thao tác quay trái hoặc quay phải (nếu tham số shift là số dương hàm sẽ tiến hành quay trái, quay phải trong trường hợp tham số shift là số âm). Tham số shift mặc định là 1. size là kích thước bao nhiêu bit mà con số value sẽ được chuyển sang số nhị phân trước khi thực hiện phép quay (mặc định là 16-bit). Size = "B" (8-bit), size = "W" (16-bit), size = "D" (32-bit).

- Phép quay trái thực hiện như thao tác dịch trái chỉ có khác ở chỗ trong các trường hợp sau :

+ Nếu Size = "B", bit số 7 được mang ra ngoài sẽ đưa vào bit 0.

+ Nếu Size = "W", bit số 15 được mang ra ngoài sẽ đưa vào bit 0.

+ Nếu Size = "D", bit số 31 được mang ra ngoài sẽ đưa vào bit 0.

- Phép quay phải thực hiện như thao tác dịch phải chỉ có khác ở chỗ trong các trường hợp sau :

+ Nếu Size = "B", bit số 0 của phép quay phải sẽ được mang vào bit số 7

+ Nếu Size = "W", bit số 0 của phép quay phải sẽ được mang vào bit số 15.

+ Nếu Size = "D", bit số 0 của phép quay phải sẽ được mang vào bit số 31.

Ví dụ : *$x = BitRotate (7, 2)*; thực hiện thao tác quay trái 2 lần

Ta có 7 (10) = 0000 0000 0000 0111 (2)

Quay lần 1 = 0000 0000 0000 1110 (2)

Quay lần 2 = 0000 0000 0001 1100 (2) ; Kết quả là 28

$y = BitRotate(14, -2); thực hiện thao tác quay phải 2 lần

Ta có 14 (10)= 0000 0000 0000 1110 (2)

Quay lần 1 = 0000 0000 0000 0111 (2)

Quay lần 2 = 1000 0000 0000 0011 (2) ; Kết quả là 32771

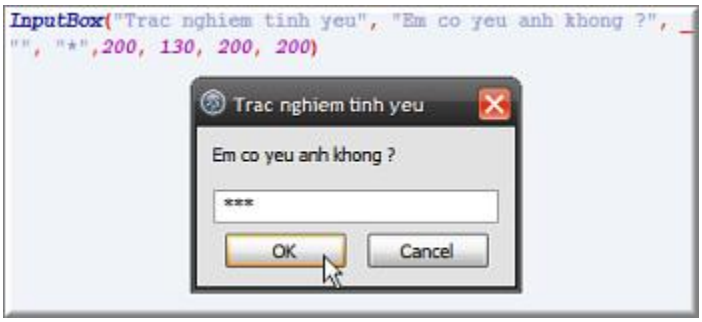
Hàm liên quan đến xử lý hộp thoại

1. InputBox("title", "prompt", ["default"], ["passchar"], [width], [height], [left], [top], [timeput])

- Công dụng : xuất hiện một hộp nhập cho người dùng gõ chuỗi thông tin vào.

- title : là tên tiêu đề cửa sổ hộp nhập. prompt : là lời nhắc gợi ý cho người dùng gõ thông tin vào. passchar : là kí tự sẽ thay thế những kí tự mà người dùng gõ vào, ví dụ nếu tham số passchar ="*", thì khi người dùng gõ kí tự vào sẽ thành dấu *. width đọ dài chiều ngang của hộp nhập. height là chiều cao của hộp nhập. left : tọa độ đỉnh trái của hộp nhập (mặc định là ở giữa màn hình). right : tọa độ đỉnh trên cùng của hộp nhập (mặc định là ở giữa màn hình). timeout : là thời gian tối đa mà hộp nhập thông tin tồn tại trên màn hình cho người dùng gõ thông tin vào, nếu quá thời gian này hộp thoại nhập sẽ biến mất.

Ví dụ :



2. MsgBox(flag, "title", "text", [timeout])

- Công dụng : xuất hiện một hộp thoại thông tin cho người dùng về vấn đề gì đó. Ví dụ như hộp thoại báo có lỗi chương trình, hoặc hộp thoại thỉnh ý người dùng có lưu tài liệu khi đóng không như trong Microsoft Word.

- flag : cờ tham số quyết định một số tùy chọn xuất hiện hộp thoại như có nút OK hay không, có biểu tượng gì hay không (bàng có thể tham khảo tham số flag trong bảng dưới đây,… title là tiêu đề cửa sổ hộp thoại. text là đoạn văn bản chứa thông tin đến người dùng. timeout là thời gian tối đa mà hộp thoại sẽ xuất hiện, qua thời gian đó hộp thoại sẽ biến mất. Tham số này mặc định là 0, khi đó hộp thoại sẽ xuất hiện mãi cho đến khi người dùng nhấn nút Close hoặc nhấn nút nhấn nào đó trên hộp thoại.

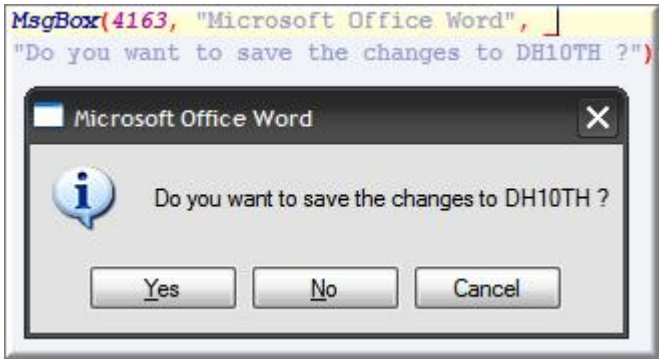
Tham số flag	**Tác động đến hộp thoại**
0	Thêm OK trên hộp thoại
1	Thêm OK và Cancel trên hộp thoại
2	Thêm Abort, Retry, và Ignore vào hộp thoại.
3	Thêm Yes, No, và Cancel vào hộp thoại.
4	Thêm Yes và No vào hộp thoại
5	Thêm Retry và Cancel vào hộp thoại
6	Thêm Cancel, Try Again, Continue vào hộp thoại
Tham số flag	**Tác động đến icon trên hộp thoại**
0	Không có icon nào xuất hiện trên hộp thoại
16	Có biểu tượng Stop
32	Có biểu tượng Question-mark
48	Có biểu tượng Exclamation-point.
64	Có biểu tượng Information-sign và bao gồm cả chữ "i"
Tham số flag	**Nút nhấn nào trên hộp thoại được bật mặc định**
0	Nút nhấn đầu tiên trên hộp thoại sẽ là nút nhấn được chọn mặc định.
256	Nút nhấn thứ hai trên hộp thoại sẽ là nút nhấn được chọn mặc định.

512	Nút nhấn thứ ba trên hộp thoại sẽ là nút nhấn được chọn mặc định.
Tham số flag	**Tác động đến kiểu dáng hộp thoại.**
0	Hộp thoại kiểu Application
4096	Hộp thoại kiểu System modal (hộp thoại có một biểu tượng)
8192	Hộp thoại kiểu Task modal

- Hàm này nếu được gọi thành công sẽ trả về số tương ứng với nút được nhấn trên hộp thoại Msgbox. Bạn có thể tham khảo bảng sau

Nút được nhấn trên hộp thoại	**Giá trị của hàm trả về sẽ là**
OK	1
CANCEL	2
ABORT	3
RETRY	4
IGNORE	5
YES	6
NO	7
TRY AGAIN	10
CONTINUE	11

Ví dụ :



Tham số flag là 4163 = 3 + 64 + 4096 (xem bảng tham số cờ flag).

3. SplashTextOn ("title", "text", [w], [h], [xpos], [ypos], [opt], ["fontname"], [fontsize])

- Công dụng : tạo một cửa sổ popup dạng text.

- Bạn xem tham số hàm trong bảng sau

title	Tiêu đề của cửa sổ popup
text	Văn bản hiển thị trong popup.
w	Tham số tùy chọn độ dài chiều ngang của cửa sổ popup (tính bằng pixel – mặc định là 500).
h	Tham số tùy chọn độ dài chiều cao của cửa sổ popup (tính bằng pixel – mặc định là 400).
xpos	Tham số tùy chọn vị trí bên trái của cửa sổ popup. Mặc định là ở giữa.
ypos	Tham số tùy chọn vị trí đỉnh trên của cửa sổ popup. Mặc định là ở giữa.
opt	1 = Đóng một khung mỏng xung quanh cửa sổ popup 2 = Loại bỏ thuộc tính luôn nổi của cửa sổ 4 = Canh trái văn bản 8 = Canh phải văn bản 16 = Cửa sổ popup có thể di chuyển 32 = Văn bản canh giữa theo chiều dọc.
fontname	Font chữ được dùng cho văn bản. Nếu bạn gõ "" sẽ dùng font chữ hệ thống (mặc định)
fontsize	Kích cỡ của font chữ, mặc định là 12.

Ví dụ :

4. SplashImageOn ("title", "file", [w], [h], [xpos], [ypos], [opt])

- Công dụng : Tạo một cửa sổ popup ảnh

- Bạn xem tham số trong bảng sau

title	Tiêu đề của cửa sổ popup ảnh
file	Đường dẫn đầy đủ đến tập tin ảnh cần làm popup (chỉ hỗ trợ định dạng BMP, GIF, JPG
w	Tham số tùy chọn độ dài chiều ngang của cửa sổ popup (tính bằng pixel – mặc định là 500).
h	Tham số tùy chọn độ dài chiều cao của cửa sổ popup (tính bằng pixel – mặc định là 400).
xpos	Tham số tùy chọn vị trí bên trái của cửa sổ popup. Mặc định là ở giữa.
ypos	Tham số tùy chọn vị trí đỉnh trên của cửa sổ popup. Mặc định là ở giữa.
opt	1 = đóng một khung mỏng xung quanh cửa sổ popup 2 = loại bỏ thuộc tính luôn nổi của cửa sổ 16 = cửa sổ popup không thể di chuyển.

Ví dụ :

5. SplashOff()

- Công dụng : tắt cửa sổ popup được tạo bởi hai hàm SplashTextOn và SplashImageOn.

6. ProgressOn("title", "maintext", "subtext", [xpos], [ypos], [opt])

- Công dụng : tạo một cửa sổ thể hiện thông tin tiến trình. Bạn có thể thấy một ví dụ của thanh tiến trình là thanh trạng thái của trình duyệt Internet Explorer thể hiện bao nhiêu phần trăm dữ liệu tải về máy của trang web mà bạn đang truy cập.

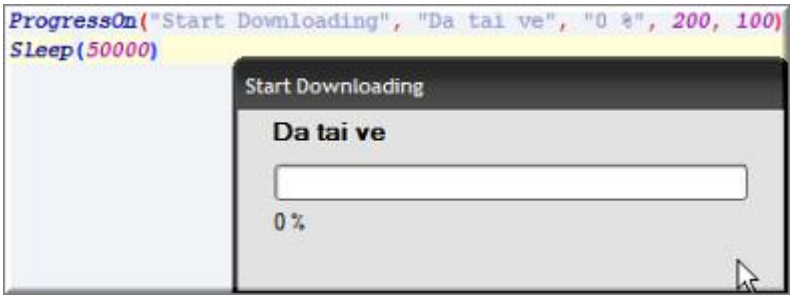
- Xem bảng thông số hàm sau

title	Tiêu đề cửa sổ thanh tiến trình.
maintext	Văn bản mà bạn muốn thiết lập trên thanh Progress.
subtext	Văn bản mà bạn muốn thiết lập dưới thanh Progress.
xpos	Tham số tùy chọn vị trí bên trái của cửa sổ popup. Mặc định là ở giữa.
ypos	Tham số tùy chọn vị trí đỉnh trên của cửa sổ popup. Mặc định là ở giữa.
opt	1 = đóng một khung mỏng xung quanh cửa sổ popup. 2 = loại bỏ thuộc tính luôn nổi của cửa sổ. 16 = cửa sổ popup không thể di chuyển.

Bạn xem ví dụ sau đây để có thể hiểu rõ hơn

Ví dụ : *ProgressOn("Start Downloading", "Da tai ve", "0 %", 200, 100)*

Sleep(50000)



Tuy nhiên hàm này chỉ mới tạo một thanh tiến trình đơn giản mà thôi. Để thiết lập một thanh tiến trình động để miêu tả quá trình, tức là đang thực hiện bao nhiêu phần trăm công việc, bạn hãy sử dụng hàm ProgressSet.

7. ProgressSet(percent, ["subtext"], ["maintext"])

- Công dụng : tạo một thanh tiến trình động thể hiện quá trình của công việc.

- percent : là tỉ lệ phần trăm mà bạn muốn gán cho thanh tiến trình (lúc này thanh tiến trình sẽ hiển thị mức độ hoàn thành công việc thông qua màu sắc của thanh tiến triển). subtext : Văn bản mà bạn muốn thiết lập dưới thanh Progress. maintext : Văn bản mà bạn muốn thiết lập trên thanh Progress.

Ví dụ :

```
ProgressOn("Start Downloading", "Da tai ve", "0 %", 135, 190)
Sleep(3000)
For $i = 5 to 100 step 5
    sleep(1000)
    ProgressSet( $i, $i & " %")
Next
ProgressSet(100 , "Done", "Complete")
```

8. ProgressOff()

- Công dụng : tắt cửa sổ thanh tiến trình được tạo bởi hàm ProgressOn.

Hàm xử lý khác

1. BlockInput(flag)

- Công dụng : có cho phép người dùng sử dụng chuột và bàn phím hay không. Nếu không cho phép, người dùng sẽ không thực hiện các thao tác chuột cũng như gõ phím.

- flag : tham số quyết định có cho phép người dùng chuột và bàn phím hay không. flag = 1, vô hiệu hóa chuột và bàn phím. flag = 0, cho phép chuột và bàn phím hoạt động.

Ví dụ : *BlockInput(1)*

Sleep(10000)

BlockInput(0)

Bạn hãy mở một chương trình soạn thảo văn bản như Microsoft Word sau đó chạy thử hàm này. Tiếp tục bạn hãy gõ một vài kí tự trên bàn phím trong cửa sổ soạn thảo Word để thấy cái hay của hàm này. Đoạn chương trình trên chỉ thực hiện vô hiệu hóa thao tác chuột và bàn phím trong vòng 10 giây.

2. Break(mode)

- Công dụng : cho phép hoặc vô hiệu hóa việc người dùng thoát khỏi đoạn mã lệnh AutoIt đang chạy trong hệ thống. (Bạn có thể dừng đoạn mã AutoIt đang chạy bằng cách nhấn chuột phải vào biểu tượng AutoIt dưới khay hệ thống, sau đó chọn Exit).

- mode : là tham số quyết định có cho phép hay không. mode =1, người dùng có thể thoát khỏi đoạn mã AutoIt đang chạy. mode = 0, người dùng phải chờ cho đoạn mã thực thi kết thúc, không thể thoát khỏi đoạn mã AutoIt đang thực hiện được. Bạn chỉ nên thiết lập mode = 0 vì một lý do đặc biệt nào đó.

3. CDTray("drive", "status")

- Công dụng : đóng hoặc mở khay ổ đĩa CD/DVD trên máy tính của bạn.

- drive : là tên ổ đĩa CD/DVD trong hệ thống của bạn cần đóng hoặc mở. status là tên trạng thái của ổ đĩa. status = "open" (mở khay ổ đĩa), status = "closed" (đóng khay ổ đĩa). Hàm này có thể không hoạt động đối với ổ CD/DVD trên laptop, mà có thể bạn phải làm nó bằng tay.

Ví dụ : *CDTray("F:", "open")*

4. VarGetType(var)

- Công dụng : trả về kiểu dữ liệu của một biến.

- var : là tên biến cần lấy thông tin kiểu dữ liệu.

Ví dụ : *$so1 = 1*

$so2 = 2.0

Msgbox(0, "Types", "$so1 is " & VarGetType($so1) & @CRLF & "$so2 is " & VarGetType($so2)); $so1 is Int32, $so2 is Double

Hàm xử lý Registry trong Windows

1. RegRead("keyname", "valuename")

- Công dụng : đọc giá trị từ một mục nào đó trong Registry

- keyname : là đường dẫn nhánh tới khóa mà bạn cần đọc thông tin. valuename : là tên mục giá trị mà bạn cần đọc. Đường dẫn nhánh trong "keyname" cần phải được bắt đầu bằng một trong những cụm từ sau : "HKEY_LOCAL_MACHINE" ("HKLM"), "HKEY_USERS" ("HKU"), "HKEY_CURRENT_USER" ("HKCU"), "HKEY_CLASSES_ROOT" ("HKCR"), "HKEY_CURRENT_CONFIG" ("HKCC")

- AutoIt hỗ trợ đọc các khóa có kiểu REG_BINARY, REG_SZ, REG_MULTI_SZ, REG_EXPAND_SZ, và REG_DWORD.

Ví dụ : *$var =*

```
RegRead("HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion
","ProgramFilesDir")
MsgBox(4096, "Program files are in:", $var)
```

Bạn có thể thay đoạn mã

RegRead("HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion", "ProgramFilesDir") thành

RegRead("HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion", "ProgramFilesDir"); cho ngắn gọn hơn.

2. RegWrite("keyname", ["valuename"], ["type"], [value])

- Công dụng : tạo một khóa hoặc ghi giá trị đến Registry.

- keyname : là khóa Registry mà bạn cần muốn ghi (tham số này nếu thiếu một thành phần trường trong đường dẫn thì mặc định nó sẽ được tạo). valuename : tên mục giá trị mà bạn cần ghi. type : loại khóa mà bạn ghi, AutoIt hỗ trợ các loại khóa như : "REG_SZ", "REG_MULTI_SZ", "REG_EXPAND_SZ", "REG_DWORD", "REG_QWORD", "REG_BINARY". value : là giá trị mà bạn muốn ghi vào mục valuename.

Ví dụ : *RegWrite("HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon", "DefaultUserName", "REG_SZ", "Mr. Trinh")*

3. RegDelete("keyname", ["valuename"])

- Công dụng : xóa một khóa hoặc một mục nào đó trong Registry.

- keyname : là tên khóa mà bạn cần xóa. valuename là tên mục nào đó trong khóa

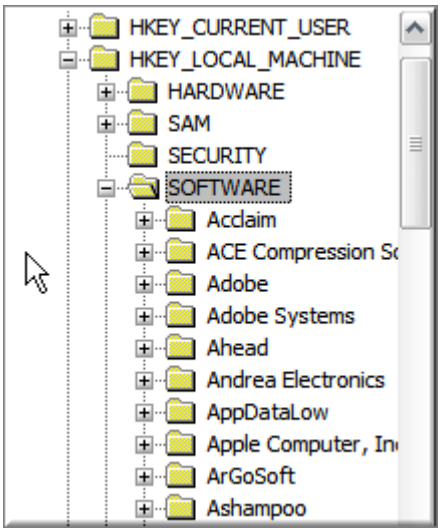
keyname mà bạn muốn xóa. Nếu hàm không có tham số valuename thì sẽ xóa khóa trong "keyname".

Ví dụ : *RegDelete("HKEY_CURRENT_USER\Software\Test", "TestKey")*

4. RegEnumKey("keyname", instance)

- Công dụng : đọc tên các khóa con trong một khóa.

- keyname : là tên khóa có chứa các khóa con mà bạn cần đọc. instance : là vị trí khóa mà bạn cần đọc trong danh sách các khóa con. Để hiểu thêm tham số instance, bạn có thể xem hình sau :



Hình trên mô tả các khóa con trong khóa SOFTWARE bao gồm : Acclaim, ACE Compression Software, Adobe, Adobe Systems, Ahead… Bây giờ nếu bạn gán instance = 1, hàm trả về Acclaim. Nếu gán về 2, hàm trả về ACE Compression Software. Nếu gán về 4, hàm trả về Adobe Systems. Bây giờ bạn có thể hiểu tham số này rồi đó.

- Hàm này nếu gọi thành công thì @error = 0.

Ví dụ :
```
For $ i= 1 to 10
        $var = RegEnumKey("HKEY_LOCAL_MACHINE\SOFTWARE", $i)
        If @error <> 0 then ExitLoop
        MsgBox(4096, "SubKey #" & $i & " under HKLM\Software: ", $var)
 Next
```

5. RegEnumVal("keyname", instance)

- Công dụng : đọc danh sách tên các mục (valuename) trong một khóa nào đó.

- keyname : là đường dẫn tới khóa mà bạn cần đọc các mục. Tham số instance tương tự như trong hàm trên RegEnumKey, nhưng áp dụng đối với mục giá trị (valuename).

Ví dụ :
```
For $i = 1 to 100
    $var = RegEnumVal("HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows", $i)
    if @error <> 0 Then ExitLoop
    MsgBox(4096, "Value Name  #" & $i & " under in Windows key", $var)
   Next
```

Hàm xử lý biến và chuyển đổi

1. Asc("char")

- Công dụng : trả về số nguyên ứng với kí tự nằm trong bảng mã ASCII.

- char là tham số bắt buộc dạng kí tự. Nếu tham số char là một chuỗi, hàm sẽ trả về chuỗi số nguyên trong bảng mã ASCII ứng với kí tự đầu tiên.

Ví dụ : *Asc("\") = 92*

2. AscW("char")

- Công dụng : trả về mã Unicode ứng với kí tự char.

- char là tham số bắt buộc dạng kí tự. Nếu tham số char là một chuỗi, hàm sẽ trả về mã Unicode của kí tự đầu tiên trong chuỗi.

Ví dụ : *MsgBox(0, "AscW", AscW("§"))*; kí tự § trong ví dụ này bạn gõ bằng cách nhấn phím Alt sau đó nhấn liên tục phím số 7, 8 và 9 trên cụm phím Numpad. Trong ví dụ này, hàm trả về 167.

3. Chr(ASCII)

- Công dụng : trả về kí tự tương ứng với tham số mã ASCII trong hàm.

- ASCII : mã ASCII cần chuyển sang kí tự.

Ví dụ : *Chr(96) = "`"*

4. ChrW(Unicode)

- Công dụng : trả về kí tự tương ứng với mã Unicode.

- Unicode : là mã Unicode cần chuyển thành kí tự.

Ví dụ : *$text = ""*

```
For $i = 678 to 900
        $text = $text & ChrW($i)
Next
MsgBox(0, "Ki tu co ma Unicode tu 678 den 900 la ", $text)
```

5. Eval("string")

- Công dụng : trả về giá trị của biến nằm trong chuỗi string.

- string : là tham số chuỗi có chứa tên biến mà bạn muốn lấy giá trị về.

Ví dụ : *Dim $tam = 12*

```
$s = Eval("tam")
MsgBox(0, "Eval", $s)
```

6. Hex(number, [length])

- Công dụng : chuyển một số nguyên đến dạng hệ thập lục phân.

- number : là số cần chuyển sang mã hexa. length : tham số tùy chọn, chứa chiều dài chuỗi kí tự của mã hexa mà hàm sẽ trả về (tối đa là 8).

Ví dụ : *$s = Hex(1991, 4); $ = 07C7*

MsgBox(0, "Eval", $s)

$s = Hex(1991, 8); $s = 000007C7

MsgBox(0, "Eval", $s)

7. Int (number)

- Công dụng : trả về phần nguyên của một con số.

- Number : là con số cần lấy phần nguyên.

Ví dụ : *Int(99.9);* = 99

8. IsAdmin()

- Công dụng : kiểm tra tài khoản người dùng hiện tại có phải là Administrator hay không.

- Hàm sẽ trả về 1 nếu tài khoản người dùng có quyền Admin, và 0 trong trường hợp ngược lại.

Ví dụ : *If IsAdmin() Then MsgBox(0, "Check Admin", "Tai khoan cua ban la Administrator")*

9. Number(expression)

- Công dụng : chuyển về số từ một biểu thức.

- expression : là biểu thức cần trả về số. Nếu expression bắt đầu bằng một kí tự hàm trả về 0. Nếu bắt đầu bằng một con số nhưng sau nó lại là một kí tự thì hàm trả về số đứng trước kí tự đó. Trong trường hợp expression là một biểu thức toán học, hàm sẽ trả về một con số tương ứng với giá trị biểu thức đó. Bạn hãy xem ví dụ sau để thấy rõ hơn hàm này.

Ví dụ : *$w = Number(1+2+10)*; trả về 13

$w = Number("1+2+10"); trả về 1

*$w = Number(8/4*9)*; trả về 18

$w = Number("9.18"); trả về 9.18

$w = Number("Mr. Trinh1991"); trả về 0.

10. String(expression)

- Công dụng : chuyển tham số expression trong hàm thành chuỗi.

11. UBound(Array, [dimension])

- Công dụng : trả về cận trên của một chiều nào đó trong mảng.

- Array : tên mảng cần lấy cận trên. dimension : là số của chiều trong mảng mảng mà bạn cần lấy cận trên. Nếu không có tham số này, mặc định là 1. Nếu tham số này = 0, hàm trả về số chiều có ở trong mảng.

Ví dụ : *Dim $myArray[10][20][30]* ;element 0,0 to 9,19

$rows = UBound($myArray); trả về 10

$cols = UBound($myArray, 2); trả về 20

$dims = UBound($myArray, 0); trả về 3

MsgBox(0, "UBound", $rows)

MsgBox(0, "UBound", $cols)

MsgBox(0, "UBound", $dims)

12. StringToBinary(expression, [flag])

- Công dụng : chuyển string sang mã hexadecimal hoặc binary.

- expression : là chuỗi cần chuyển. flag : tham số quy định kiểu định dạng hexadecimal xuất ra, mặc định là 1. flag = 1, định dạng theo mã ANSI. flag = 2, định dạng theo mã UTF16 Little Endian. flag = 3, định dạng theo mã UTF16 Big Endian. flag = 4, định dạng theo mã UTF8.

Ví dụ : *$a = StringToBinary("DH10TH")*; trả về 0X444831305448

MsgBox(0, "StringToBinary", $a)

$a = StringToBinary("DH10TH", 2); trả về 0X4400480031003000540048 00

MsgBox(0, "StringToBinary", $a)

$a = StringToBinary("DH10TH", 3); trả về 0X004400480031003000540048

MsgBox(0, "StringToBinary", $a)

$a = StringToBinary("DH10TH", 4); trả về 0X444831305448

MsgBox(0, "StringToBinary", $a)

13. IsArray(var)

- Công dụng : kiểm tra một biến có phải là kiểu dữ liệu mảng hay không. Trả về 1 nếu đúng, trả về 0 nếu không phải kiểu mảng.

14. IsBinary(expression)

- Công dụng : kiểm tra một biến và biểu thức có phải là kiểu mã hexadecimal hoặc số nhị phân hay không.

- expression : là biến hoặc biểu thức cần kiểm tra. Trả về 1 nếu đúng, trả về 0 nếu sai.

Ví dụ : *$bin = Binary("0x00204060")*

$str = "0x00204060"

msgbox(0, "IsBinary $bin", IsBinary($bin)); trả về 1

msgbox(0, "IsBinary $str", IsBinary($str)); trả về 0 bởi vì $str là kiểu dữ liệu là string. Muốn nó là kiểu hexadecimal cần phải chuyển nó thông qua hàm Binary tương tự như $bin.

15. IsBool(var)

- Công dụng : kiểm tra một biến có phải kiểu dữ liệu boolean hay không. Trả về 1 nếu đúng, trả về 0 trong trường hợp ngược lại.

16. IsDeclared(var)

- Công dụng : kiểm tra một biến nào đó có được gán giá trị chưa.

- var là tên biến cần kiểm tra. Bỏ dấu $ trước tên biến cần kiểm tra.

Ví dụ : *If Not IsDeclared ("a") then*

MsgBox(0,"", "$a la khong duoc gan gia tri")

```
EndIf
$a=1
If IsDeclared ("a") then
        MsgBox(0,"", "$a duoc gan gia tri"  )
EndIf
```

17. IsFloat(var)

- Công dụng : Kiểm tra một biến hoặc biểu thức có phải kiểu số thực hay không.

- var là biến hoặc biểu thức cần kiểm tra.

Ví dụ : *IsFloat(3.14159)* ;trả về 1

IsFloat(3.000) ; trả về 0 vì bản chất 3.000 là một con số nguyên.

IsFloat(1/2 - 5) ; trả về 1

IsFloat(1.5e3) ; trả về 0 bởi vì 1.5e3 = 1.5 * 1000 = 1500.

IsFloat("12.345") ; trả về 0 bởi vì nó là một chuỗi.

17. IsInt(var)

- Công dụng : kiểm tra một biến hoặc một biểu thức có phải là kiểu số nguyên hay không.

- var là tên biến hoặc biểu thức cần kiểm tra.

Ví dụ : *IsInt(-12345)* ; trả về 1

IsInt(3.0000) ; trả về 1

IsInt("5432") ; trả về 0 vì đây là một chuỗi

IsInt(7.5 - 4.5) ; trả về 1 bởi vì biểu thức 7.5 – 4.5 = 3, là một con số nguyên.

18. IsKeyword(var)

- Công dụng : Kiểm tra một biến có phải là chứa từ khóa nào đó hay không.

- var : là tên biến cần kiểm tra.

Ví dụ : *$a = default*

```
If IsKeyword($a) Then MsgBox(0,"Ok", "Yes it is")
```

19. IsNumber(var)

- Công dụng : kiểm tra một biến có phải là kiểu số hay không.

- var là tên biến cần kiểm tra.

Ví dụ : *$a = 3.5*

IsNumber($a); true

$a = 3/4;

IsNumber($a); true

20. Binary(expression)

- Công dụng : chuyển biểu thức expression sang kiểu mã hexadecimal hoặc số nhị phân.

21. BinaryLen(binary)

- Công dụng : trả về số byte của số hexadecimal hoặc số nhị phân.

- binary : là biến hoặc số hexadecimal hay số dạng nhị phân.

22. Dec("hex")

- Công dụng : chuyển số hexadecmail (hệ thập lục phân) sang hệ thập phân.

- hex : chuỗi chứa số thập lục phân cần chuyển đổi.

Ví dụ : *$hex = Dec("34FF")*; trả về 13567

MsgBox(0, "Dec Test", $hex)

23. BinaryMid(binary, start, [count])

- Công dụng : lấy một phần dữ liệu của số nhị phân hoặc số hexadecimal.

- binary : là số nhị phân hoặc số hexadecimal cần lấy. start : là byte bắt đầu cần lấy. count là số byte cần lấy, tính tử vị trí start.

Ví dụ : *$binary = Binary("0x10203040")*

$extract = BinaryMid($binary, 2, 2); trả về 0x2030

MsgBox(0, "Byte 2 và Byte 3 la", $extract)

Hàm thời gian

1. Sleep(delay)

- Công dụng : tạm dừng thời gian thực thi đoạn mã của chương trình AutoIt trong khoản thời gian nào đó.

- delay : là thời gian cần trì hoãn, tính bằng ms.

Ví dụ : *Sleep(4000)*; tạm dừng chương trình 4 giây.

2. TimerInit() và TimerDiff(timestamp)

- Công dụng : hàm TimerInit trả về một timestamp tương ứng với thời gian hiện tại đúng vào lúc gọi hàm này. TimerDiff sẽ tính khoảng thời gian lúc gọi hàm TimeInit đến thời gian gọi hàm TimerDiff.

Ví dụ : *$begin = TimerInit()*

```
sleep(3000)
$dif = TimerDiff($begin)
MsgBox(0,"Khoang thoi gian la ",$dif)
```

Trong AutoIt có một số macro về thời gian bạn có thể tham khảo bảng sau :

Macro	Mô tả
@MSEC	Số ms hiện tại của đồng hồ hệ thống. Có giá trị từ 00 đến 999.
@SEC	Giây hiện tại của đồng hồ hệ thống. Có giá trị từ 00 đến 59

@MIN	Phút hiện tại của đồng hồ hệ thống. Có giá trị từ 00 đến 59
@HOUR	Giờ hiện tại của hệ thống. Có giá trị từ 00 đến 23
@MDAY	Ngày hiện tại của tháng. Có giá trị từ 01 đến 31
@MON	Tháng hiện tại trong năm. Có giá trị từ 01 đến 12
@YEAR	Năm hiện tại (bốn chữ số)
@WDAY	Số chỉ thứ trong tuần có giá trị từ 1 đến 7. 1 (Chủ nhật), 2 (Thứ hai), 3 (Thứ ba),…
@YDAY	Ngày thứ mấy trong năm. Có giá trị từ 001 đến 366 hoặc 001 đến 365 nếu không phải là năm nhuận.

Hàm quản lý cửa sổ

1. WinActivate("title")

- Công dụng : Kích hoạt một cửa sổ của chương trình nào đó.

- Title là tên tiêu đề cửa sổ cần kích hoạt. Hàm này sẽ trả về 0 nếu tiêu đề cửa sổ trong hàm không tìm thấy hoặc không thể kích hoạt được cửa sổ đó.

Ví dụ : *WinActivate("Untitled – Notepad")*

2. WinActive("title")

- Công dụng : kiểm tra xem một cửa sổ có tồn tại và đang ở trong trạng thái kích hoạt hay không.

- title : là tên tiêu đề cửa sổ cần kiểm tra. Hàm này sẽ trả về 0 nếu cửa sổ đó không có tồn tại hoặc không ở trạng thái kích hoạt.

Ví dụ : *If WinActive("Untitled – Notepad") Then*

MsgBox(0, "WinActive", "Cua so Notepad hien duoc kich hoat")

EndIf

3. WinClose("title")

- Công dụng : đóng một cửa sổ đang mở.

- title là tên tiêu đề cửa sổ cần đóng.

Ví dụ : *WinClose("Untitled – Notepad")*

4. WinExists("title")

- Công dụng : kiểm tra xem một cửa sổ chương trình có tồn tại hay không.

- title : là tên tiêu đề cửa sổ cần kiểm tra.

Ví dụ : *WinExists("Untitled – Notepad")*

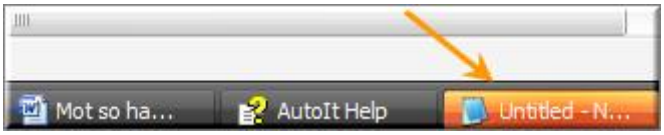
5. WinFlash("title", ["text"], [flash], [delay])

- Công dụng : tạo hiệu ứng nhấp nháy thanh chương trình trên taskbar.

- Bảng chú giải tham số

title	Là tên tiêu đề cửa sổ cần làm hiệu ứng.
text	Tham số tùy chọn là văn bản trong cửa sổ cần làm hiệu ứng. Bạn nên để tham số này là "".
flash	Số hiệu ứng nhấp nháy trên cửa sổ. Mặc định là 4
delay	Thời gian giữa các lần làm hiệu ứng nhấp nháy. Mặc định là 500ms.

Ví dụ : *WinFlash("Untitled – Notepad", "", 6)*



6. WinGetClientSize("title")

- Công dụng : lấy kích thước của cửa sổ một chương trình.

- title : là tiêu đề cửa sổ cần lấy kích thước. Hàm sẽ trả về một mảng có hai phần tử. Trong đó phần tử thứ nhất chứa độ dài chiều ngang của cửa sổ, phần tử thứ hai chứa độ dài chiều cao cửa sổ. Nếu cửa sổ cần lấy thông tin kích thước bị thu nhỏ trên khay hệ thống, hàm sẽ trả về 0.

- Nếu muốn lấy thông tin kích thước của cửa sổ desktop của bạn, hãy cho tham số title = "Program Manager".

Ví dụ : *$a = WinGetClientSize("Program Manager")*

MsgBox(0, "WinGetClientSize", $a[0] & " " & $a[1])

7. WinGetPos("title")

- Công dụng : trả về thông tin vị trí và kích thước của cửa sổ được cho.

- title : là tên tiêu đề cửa sổ cần lấy thông tin. Hàm này nếu gọi thành công sẽ trả về bốn phần tử. Trong đó phân tử thứ nhất và hai chứa tọa độ X, Y của góc ở đỉnh trên cùng bên trái cửa sổ. Phần tử thứ ba, tư chứa thông tin kích thước của cửa sổ (tương tự WinGetClientSize).

Ví dụ : *$a = WinGetPos("Untitled – Notepad")*

MsgBox(0, "WinGetClientSize", $a[0] & " " & $a[1])

8. WinGetProcess("title")

- Công dụng : lấy thông tin mã số process (PID) của một cửa sổ có tiêu đề title.

9. WinGetState("title")

- Công dụng : trả về thông tin trạng thái của cửa sổ.

- Hàm được gọi thành công sẽ trả về các số tương ứng với các trạng thái của cửa sổ : 1 (cửa sổ có tồn tại), 2 (cửa sổ là hiện hữu cho người dùng thấy), 4 (cửa sổ khả dụng), 8 (cửa sổ đang kích hoạt), 16 (cửa sổ đang thu nhỏ trên khay hệ thống), 32 (cửa sổ đang ở trạng thái phóng lớn toàn màn hình).

10. WinGetText("title")

- Công dụng : lấy chuỗi văn bản được chứa trong một cửa sổ chương trình.

- title : là tên tiêu đề cửa sổ cần lấy văn bản. Hàm này có thể lấy về văn bản có dung lượng tối đa là 64KB và chỉ làm việc với các cửa sổ chương trình được thu nhỏ trên khay hệ thống.

Ví dụ :

11. WinKill("title")

- Công dụng : đóng một cửa sổ. Chúng ta đã biết hàm WinClose cũng có tính năng tương tự nhưng điểm khác biệt giữa hai hàm này là ở chỗ WinClose là đóng mềm (hàm WinClose tương tự như thao tác ta nhấn nút Close trên cửa sổ chương trình), còn WinKill là đóng cứng tức là nếu chương trình có lỗi gì hoặc phải lưu thao tác gì đó trong cửa sổ chương trình thì hàm WinKill sẽ đóng ngay lập tức không cho người dùng kịp lưu phiên làm việc. Cho nên bạn nên cần thận khi sử dụng hàm này vì có thể bạn không thể lưu công việc hiện thời trong cửa sổ chương trình đang làm việc.

- title : là tên tiêu đề cửa sổ cần đóng.

Ví dụ : *WinKill("Untitled – Notepad")*

Bạn có thể so sánh điểm khác biệt giữa hàm WinKill và WinClose bằng cách trong cửa sổ Notepad, bạn gõ một nội dung văn bản nào đó. Sau đó lần lượt dùng hàm WinClose và WinKill để thấy rõ sự khác biệt.

12. WinMinimizeAll()

- Công dụng : thu nhỏ các sổ chương trình đang mở xuống khay hệ thống.

13. WinMinimizeAllUndo()

- Công dụng : hủy thao tác gọi hàm WinMinimizeAll()

14. WinMove("title", "text", x, y, [width], [height], [speed])

- Công dụng : di chuyển cửa sổ chương trình từ vị trí này sang vị trí khác. Hàm này còn có công dụng resize lại kích thước cửa sổ.

- Xem thông số hàm

title	Tiêu đề của cửa sổ cần di chuyển hoặc resize.
text	Đoạn văn bản chứa trong cửa sổ cần di chuyển hoặc resize. Nên để tham số này là "" bởi vì chỉ cần có tham số title là đủ.
X	Tọa độ X mà cửa sổ cần di chuyển tới
y	Tọa độ Y mà cửa sổ cần di chuyển tới
width	Kích thước chiều ngang mới của cửa sổ
height	Kích thước chiều cao mới của cửa sổ
speed	Tốc độ di chuyển cửa sổ. Nó có giá trị từ 1 (nhanh nhất) đến 100 (chậm nhất). Nếu không định nghĩa tham số này, cửa sổ chương trình sẽ di chuyển tức thì.

- Ghi chú : hàm này không có tác dụng đối với các cửa sổ thu nhỏ trên thanh Taskbar.

Ví dụ : *WinMove("Untitled – Notepad", 0, 0, 200, 200)*

15. WinSetState("title", "text", flag)

- Công dụng : thiết lập trạng thái của một cửa sổ.

- Xem bảng thống số hàm

title	Tiêu đề của cửa sổ cần thiết lập trạng thái.
text	Văn bản chứa trong cửa sổ cần thiết lập trạng thái. Bạn nên để tham số này là "".
flag	Tham số flag chỉ trạng thái của cửa sổ. @SW_HIDE = ẩn cửa sổ chương trình @SW_SHOW = hiện cửa sổ chương trình đã bị ẩn trước đó. @SW_MINIMIZE = thu nhỏ cửa sổ chương trình trên thanh Taskbar. @SW_MAXIMIZE = phóng lớn cửa sổ chương trình toàn màn hình. @SW_RESTORE = khôi phục trạng thái của cửa sổ đến trạng thái Normal. @SW_DISABLE = đóng băng cửa sổ chương trình (cửa sổ không thể di chuyển hoặc thay đổi trạng thái của cửa sổ). @SW_ENABLE = phục hồi cửa sổ chương trình bị đóng băng.

Ví dụ : *WinSetState("Untitled – Notepad", "", @SW_DISABLE)*

Sleep(7000)

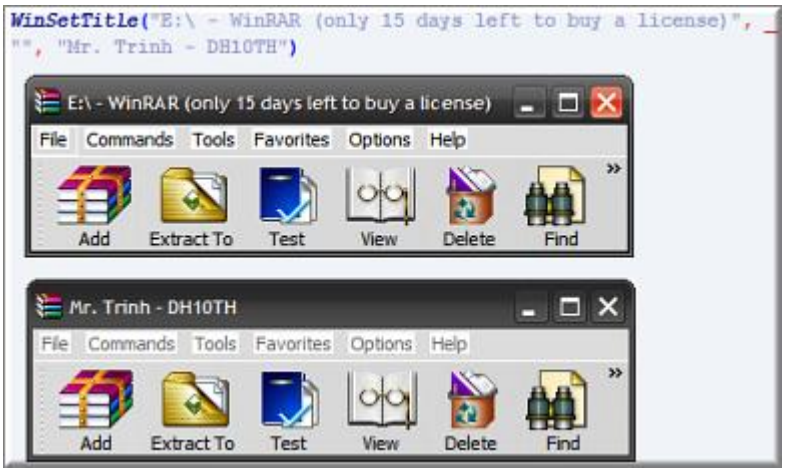
WinSetState("Untitled – Notepad", "", @SW_ENABLE)

16. WinSetTitle("title", "text", "newtitle")

- Công dụng : thay đổi tiêu đề của một cửa sổ chương trình.

- title : tên tiêu đề cửa sổ cần thay đổi. Tham số text bạn nên để "", newtitle là tên tiêu đề mới của cửa sổ.

Ví dụ :



17. WinSetTrans("title", "text", transparency)

- Công dụng : thiết lập độ mờ của một cửa sổ.

- title : là tên tiêu đề cửa sổ cần thiết lập. text nên để "". transparency : là con số quy định độ mờ, có giá trị từ 0 đến 255. Tham số này càng dần về 255 thì cửa sổ càng sẽ nhìn thấy rõ, còn về đến 0 thì cửa sổ hết thấy luôn.

Ví dụ : *WinSetTrans("Untitled – Notepad", "", 67)*

18. WinWait("title", ["text"], [timeout])

- Công dụng : tạm dừng đoạn mã lập trình AutoIt đang thực thi cho đến khi cửa sổ nào đó tồn tại. Hay nói cách khác, đoạn mã lệnh AutoIt sau lệnh gọi hàm WinWait sẽ được thực thi nếu có cửa sổ yêu cầu xuất hiện.

- title : là tên tiêu đề cửa sổ. Tham số text nên để "", timeout (tính bằng giây) là thời gian tối đa mà hàm WinWait có thể thực thi, tức là sau khoảng thời gian này đoạn mã lệnh sau lệnh gọi hàm WinWait sẽ được thực thi cho dù cửa sổ chương trình có tham số title có xuất hiện hay không.

Ví dụ : *WinWait("Untitled – Notepad", "", 8)*

MsgBox(0, "Check WinWait", "Hello World")

19. WinWaitActive("title", ["text"], [timeout])

- Công dụng : tạm dừng đoạn mã chương trình đang thực thi cho đến khi cửa sổ nào đó được kích hoạt.

- Ý nghĩa tham số tương tự như hàm WinWait.

20. WinWaitClose("title", ["text"], [timeout])

- Công dụng : tạm dừng đoạn mã chương trình đang thực thi cho đến khi cửa sổ nào đó bị đóng.

- Ý nghĩa tham số tương tự như hàm WinWait.

21. WinWaitNotActive("title", ["text"], [timeout])

- Công dụng : tạm dừng đoạn mã chương trình đang thực thi cho đến khi cửa sổ nào đó không ở trạng thái kích hoạt.

- Ý nghĩa tham số tương tự như hàm WinWait.

Hàm xử lý chuỗi

1. StringCompare("string1", "string2", [flag])

- Công dụng : so sánh hai chuỗi.

- string1 : là chuỗi thứ nhất, string2 là chuỗi thứ hai cần so sánh. flag là tham số để quyết định cách so sánh hai chuỗi. Nếu flag = 0 (mặc định), khi so sánh không phân biệt chữ hoa chữ thường của cùng kí tự. flag = 1, so sánh có phân biệt chữ hoa, chữ thường của cùng kí tự. flag = 2, so sánh không phân biệt chữ hoa chữ thường nhưng dùng phương thức so sánh cơ bản trong AutoIt.

- Hàm sẽ trả về 0 nếu string1 và string2 bằng nhau. Trả về số lớn hơn 0 nếu string1 lớn hơn string2, trả về số nhỏ hơn 0 nếu string1 nhỏ hơn string2.

Ví dụ : *$result = StringCompare("Le Vien Trinh", "le vien trinh", 0)*; trả về 0

MsgBox(0, "Ket qua ham result so sanh lan 1", $result)

$result = StringCompare("Le Vien Trinh", "le vien trinh", 1); trả về -1

MsgBox(0, "Ket qua ham result so sanh lan 2", $result)

$result = StringCompare("Le Vien Trinh", "le vien trinh", 2); trả về 0

MsgBox(0, "Ket qua ham result so sanh lan 3", $result)

2. StringInStr("string", "substring", [flag], [occur], [start], [count])

- Công dụng : tìm vị trí của một chuỗi con trong một chuỗi khác.

- Bạn xem tham số hàm trong bảng sau

string	Chuỗi lớn chứa chuỗi con cần tìm.
substring	Chuỗi con cần tìm trong chuỗi string.
flag	Tham số quyết định có so sánh chữ hoa, chữ thường khi tìm vị trí của chuỗi substring trong chuỗi string hay không. 0 = không so sánh chữ hoa, chữ chường. 1 = so sánh chữ hoa, chữ thường. 2 = so sánh không phân biệt chữ hoa chữ thường nhưng dùng phương thức so sánh cơ bản trong AutoIt.

occur	Tham số tùy chọn. Tham số này quy định kết quả thứ mấy được lấy trong số các kết quả tìm kiếm được trả về. Nếu muốn quy định lấy kết quả thứ mấy từ phải qua trái, bạn hãy dùng số âm để đặt cho tham số này. Mặc định của tham số này là 1, tức là kết quả đầu tiền trong số kết quả trả về (tính từ trái qua phải). Để hiểu tham số này, bạn hãy xem ví dụ sau : $location = StringInStr("How much wood could a woodchuck chuck is a woodchuck could chuck wood?", "wood", 0, 2). Đoạn mã lệnh này tìm chuỗi "wood" trong chuỗi "How much…wood". Ta thấy sẽ có bốn kết quả trả về (vì có 4 từ wood trong chuỗi trên) tương ứng với vị trí thứ 10, 23, 44, 66. Tuy nhiên trong mã lệnh trên ta quy định tham số occur là 2, cho nên kết quả 23 sẽ được trả về. Nếu tham số occur là -1, hàm sẽ trả về 66 (vị trí thứ nhất tính từ phải qua trái).
start	Tham số tùy chọn quy định vị trí bắt đầu trong chuỗi string cần tìm kiếm.
count	Tham số tùy chọn quy định số kí tự cần tìm kiếm trong chuỗi string để so sánh với chuỗi substring. Trong đoạn mã $location = StringInStr("How much wood could a woodchuck chuck is a woodchuck could chuck wood?", "wood", 0, 2, 2, 10). Ta có tham số start = 2, count = 10, thì nó sẽ bắt đầu tìm kiếm tại chữ o (trong chữ How) cho đến chữ o trong chữ wood (sau chữ much). Tham số này phải lớn hơn độ dài chuỗi substring.

Ví dụ : *$location = StringInStr("lvt lv dttl", "lv", 0, 1, 2, 6)* ; trả về 5

MsgBox(0, "", $location)

3. StringIsAlNum("string")

- Công dụng : kiểm tra một chuỗi có phải là con số hay không.

- string là chuỗi cần kiểm tra. Hàm sẽ trả về 1 nếu chuỗi là số, còn trả về 0 trong trường hợp ngược lại.

Ví dụ : *$x = "Le Vien Trinh – 1991"*; trả về 0

MsgBox(0,"StringIsAlNum", StringIsAlNum($x))

4. StringIsAlpha("string")

- Công dụng : kiểm tra chuỗi có phải chỉ chứa các kí tự hay không

- string : là chuỗi cần kiểm tra. Hàm sẽ trả về 1 nếu chuỗi toàn là kí tự, còn trả về 0 trong trường hợp ngược lại. Tuy nhiên nếu chuỗi string có chứa khoảng trắng hàm vẫn trả về 0.

Ví dụ : *$x = "LeVienTrinh"*

MsgBox(0,"StringIsAlNum", StringIsAlpha($x)); trả về 1

5. StringIsASCII("string")

- Công dụng : kiểm tra một chuỗi có chứa kí tự chỉ nằm trong bảng mã ASCII hay không.

- string là chuỗi cần kiểm tra. Hàm trả về 1 nếu nó chỉ chứa các kí tự nằm trong bảng

mã ASCII, 0 trong trường hợp ngược lại.

Ví dụ : *$x = "LeVienTrinh"*

MsgBox(0,"StringIsAlNum", StringIsASCII($x)); trả về 1

6. StringIsDigit("string")

- Công dụng : kiểm tra xem một chuỗi có phải chỉ chứa các số từ 0 đến 9 không.

- string là chuỗi cần kiểm tra. Hàm trả về 1 nếu nó đúng, trả về 0 trong trường hợp sai.

Ví dụ : *StringIsDigit("12333")*; trả về 1

StringIsDigit("1.5"); trả về 0 vì có chứa dấu chấm
StringIsDigit("1 2 3"); trả về 0 vì nó có chứa khoảng trắng.

StringIsDigit(""); trả về 0

7. StringIsFloat("string")

- Công dụng : kiểm tra chuỗi string có phải là một con số thực hay không. Trả về 1 nếu đúng, trả về 0 nếu sai.

8. StringIsInt("string")

- Công dụng : kiểm tra chuỗi string có phải là một con số nguyên hay không. Trả về 1 nếu đúng, trả về 0 nếu sai.

9. StringIsLower("string")

- Công dụng : kiểm tra chuỗi string có phải chứa các chữ cái ở dạng chữ thường hay không. Trả về 1 nếu đúng, trả về 0 nếu sai.

10. StringIsSpace("string")

- Công dụng : kiểm tra chuỗi string có phải chỉ chứa các khoảng trắng hay không. Trả về 1 nếu đúng, trả về 0 nếu sai.

11. StringIsUpper("string")

- Công dụng : kiểm tra chuỗi string có phải chứa các chữ cái ở dạng chữ hoa hay không. Trả về 1 nếu đúng, trả về 0 nếu sai.

12. StringIsXDigit("string")

- Công dụng : kiểm tra chuỗi string có phải chỉ chứa các chữ cái và số dạng hexadecimal (0 – 9, A – F) không. Trả về 1 nếu đúng, trả về 0 nếu sai.

13. StringLeft("string", count)

- Công dụng : trả về số kí tự con nằm trong chuỗi string, tính từ bên trái qua phải.

- string : là chuỗi cần thực hiện. count là số kí tự cần lấy.

Ví dụ : *$result = StringLeft("DH10TH – Dai hoc An Giang", 6)*; trả về DH10TH

MsgBox(0, "StringLeft", $result)

14. StringLen("string")

- Công dụng : trả về số kí tự có trong chuỗi.

15. StringLower("string")

- Công dụng : chuyển đổi tất cả chữ cái trong string sang chữ thường.

16. StringMid("string", start, [count])

- Công dụng : lấy chuỗi con trong chuỗi string.

- start : vị trí bắt đầu lấy. count : tham số tùy chọn chỉ số kí tự cần lấy, bắt đầu từ start. Nếu không có tham số này, những kí tự còn lại bắt đầu từ start sẽ được trả về.

Ví dụ : *$result = StringMid("DH10TH – Dai hoc An Giang", 10, 7)*; trả về "Dai hoc"

MsgBox(0, "StringLeft", $result)

$result = StringMid("DH10TH – Dai hoc An Giang", 10); trả về "Dai hoc An Giang"

MsgBox(0, "StringLeft", $result)

17. StringUpper("string")

- Công dụng : chuyển tất cả kí tự trong chuỗi string sang chữ hoa.

18. StringTrimLeft("string", count)

- Công dụng : loại bỏ số kí tự bên trái của chuỗi string

- string : là chuỗi cần tiến hành loại bỏ. count là số kí tự bên trái cần loại bỏ.

Ví dụ : *$result = StringTrimLeft("Le Vien Trinh", 3)*; trả về "Vien Trinh"

MsgBox(0, "StringTrimLeft", $result)

19. StringTrimRight("string", count)

- Công dụng : loại bỏ số kí tự bên phải của chuỗi string

- string : là chuỗi cần tiến hành loại bỏ. count là số kí tự bên phải cần loại bỏ.

Ví dụ : *$result = StringTrimRight("Viet Nam que huong toi", 14)*; trả về "Viet Nam"

MsgBox(0, "StringTrimLeft", $result)

20. StringRight("string", count)

- Công dụng : trả về số kí tự con nằm trong chuỗi string, tính từ bên phải qua trái.

- string : là chuỗi cần thực hiện. count là số kí tự cần lấy.

Ví dụ : *$result = StringLeft("DH10TH – Dai hoc An Giang", 16)*; trả về "Dai hoc An Giang"

MsgBox(0, "StringLeft", $result)

21. StringToASCIIArray("string", [start], [end], [encoding])

- Công dụng : chuyển đổi các kí tự trong chuỗi đến một mảng chứa các mã ASCII của từng kí tự đó.

- string : là chuỗi cần chuyển đổi. start là vị trí kí tự bắt đầu của chuỗi để tiến hành chuyển đổi (mặc đinh là 0). end là số kí tự cần chuyển đổi sang mã ASCII tính từ vị trí start (mặc định là StrLen("string")). encoding : tham số tùy chọn chỉ định định dạng mã hóa kết quả trả về. 0 (UTF-16) mặc định, 1 (ANSI), 2(UTF-8).

Ví dụ : *$a = StringToASCIIArray("DH10TH")*

$msg = ""

For $i=0 To Ubound($a, 1)-1 Step 1

$msg = $msg& " "& $a[$i]

Next; msg = "68 72 49 48 84 72"

MsgBox(0, "StringToASCIIArray", $msg)

22. StringStripCR("string")

- Công dụng : loại bỏ tất cả các kí tự về đầu dòng CR (CHR(13)) của một chuỗi.

- string : là chuỗi cần gỡ bỏ.

Ví dụ : *$result = StringStripCR("I am" & Chr(13) & " Trinh")*

MsgBox(0, "Chuoi sau khi bo", $result)

23. StringStripWS("string", flag)

- Công dụng : loại bỏ các khoảng trắng trong một chuỗi.

- string là chuỗi cần loại bỏ khoảng trắng. Bạn xem tham số flag trong bảng sau

flag	Tham số này quy định cách thức xóa khoảng trắng trong chuỗi. 1. Loại bỏ khoảng trắng ở đầu chuỗi. 2. Loại bỏ khoảng trắng ở cuối chuỗi 4. Loại bỏ các khoảng trắng khi số khoảng trắng từ hai trở lên (liền nhau) giữa các từ. Khi đó chỉ còn một khoảng trắng giữa các từ. 8. Gỡ bỏ tất cả khoảng trắng trong chuỗi. Bạn có thể áp dụng nhiều phương thức xóa bằng cách cộng các cờ flag mà bạn muốn. Ví dụ bạn muốn xóa khoảng trắng đầu chuỗi và cuối chuỗi, bạn cho tham số flag = 3 (1+2).

Ví dụ : *$result = "Dai hoc An Giang"*

MsgBox(0, "Chuoi sau khi bo khoang trang", StringStripWS($result, 8)); trả về "DaihocAnGiang"

24. StringFromASCIIArray(array, [start], [end], [encoding])

- Công dụng : chuyển đổi mã code ASCII nằm trong một mảng sang chuỗi string tương ứng với các mã ASCII đó.

- Bảng thông số hàm

array	Biến mảng có các phần tử là các mã ASCII cần chuyển đổi.
start	Vị trí phần tử bắt đầu trong mảng cần chuyển đổi. Mặc định là 0.
end	Số phần tử trong mảng cần chuyển đổi, tính từ vị trí start. Mặc định là Ubound($array) – 1).

encoding	Nếu mảng chứa các phần tử ASCII cần chuyển đổi của bạn được encoding theo các mã như UTF-16, ANSI, UTF-8 thì bạn nên quy định tham số encoding như sau : - 0 : UTF – 16 (mặc định) - 1 : ANSI - 2 : UTF – 8

Ví dụ : *Dim $array[4] = [68, 84, 84, 76]*

MsgBox(0, "Chuoi sau khi chuyen doi", StringFromASCIIArray($array, 0, 4)); trả về DTTL

25. StringReplace("string", "searchstring", "replacestring", [occur], [case])

- Công dụng : thay đổi nội dung của một chuỗi con nào đó trong chuỗi.

- Bảng thông số hàm

string	Chuỗi có nội dung cần thay đổi.
searchstring/start	Chuỗi con trong chuỗi string cần tìm kiếm để thay thế thành một chuỗi khác hoặc vị trí bắt đầu trong chuỗi string cần tiến hành thay thế.
replacestring	Chuỗi thay thế chuỗi searchstring/start trong chuỗi string.
occurr	- Tham số tùy chọn. Tham số này quy định chuỗi thứ mấy sẽ được thay thế trong số các kết quả tìm kiếm được trả về. Nếu muốn quy định chuỗi thứ mấy được thay thế tính từ phải qua trái, bạn hãy dùng số âm để đặt cho tham số này. Mặc định của tham số này là 1, tức là kết quả đầu tiên trong số kết quả sẽ được thay thế (tính từ trái qua phải). Để hiểu tham số này, bạn hãy xem ví dụ sau : StringReplace("How much wood could a woodchuck chuck is a woodchuck could chuck wood?", "wood", "water", 2). Đoạn mã lệnh này thay chuỗi "wood" thành chuỗi "water" trong chuỗi "How much…wood". Ta thấy sẽ có bốn từ "wood" cần được thay thế (vì có 4 từ wood trong chuỗi trên) tương ứng với vị trí thứ 10, 23, 44, 66. Tuy nhiên trong mã lệnh trên ta quy định tham số occur là 2, cho nên từ "wood" trong "woodchuck" sẽ được thay thế thành water. Nếu tham số occur là -1, từ "wood" cuối cùng trong câu (trước dấu ?) sẽ được thay thế, các từ "wood" còn lại trong câu vẫn được giữ nguyên. - Nếu bạn quy định tham số này = 0, hàm sẽ thay thế tất cả các chuỗi cần được thay thế tìm thấy. Trong ví dụ trên, nếu bạn quy định tham số occur = 0, bốn từ wood trong câu sẽ được thay thế thành water.
case	0 = không so sánh chữ hoa, chữ chường. 1 = so sánh chữ hoa, chữ thường. 2 = so sánh không phân biệt chữ hoa chữ thường nhưng dùng phương thức so sánh cơ bản trong AutoIt.

- Hàm này nếu gọi thành công sẽ trả về một chuỗi. Số chuỗi thay thế sẽ được lưu trữ trong macro @extended.

Ví dụ : *$text = StringReplace("How much wood could a woodchuck chuck is a woodchuck could chuck wood ?", 2, "water")*; trả về "Hwaterch wood could a woodchuck chuck is a woodchuck could chuck wood ?

MsgBox(0, "StringReplace", $text)

26. StringAddCR("string")

- Công dụng : thêm kí tự về đầu dòng (Chr(13) - @CR) trước các kí tự xuống dòng (Chr(10) - @LF) trong chuỗi.

- string là chuỗi kí tự cần thêm @CR.

Ví dụ : *$old = "Notepad" & @LF & "expects" & @LF & "CRLF text."*

$new = StringAddCR($old)

27. StringSplit("string", "del", [flag])

- Công dụng : tách một chuỗi thành các chuỗi con tùy thuộc vào kí tự phân cách chuỗi trong tham số del của hàm. Hàm này nếu gọi thành công sẽ trả về một mảng trong đó phần tử đầu tiên chứa số chuỗi kí tự được tách ra, các phần tử còn lại tương ứng với các chuỗi kí tự sau khi được tách ra từ chuỗi gốc. (Bạn hãy xem ví dụ mô tả phía dưới để hiểu thêm ý nghĩa của hàm này).

- string : là chuỗi mà bạn muốn tách thành các chuỗi con. del là tham số chứa chuỗi kí tự cần làm cơ sở để phân tách chuỗi gốc thành các chuỗi con (có phân biệt chữ thường, chữ hoa). flag là tham số quy định cách thức tách chuỗi dựa vào chuỗi kí tự trong tham số del (mặc định là 0). Nếu tham số này là 0, mỗi kí tự trong tham số del sẽ làm cơ sở để tách chuỗi. Nếu tham số này là 1, toàn bộ kí tự trong tham số del sẽ là cơ sở để tách chuỗi. Nếu tham số này là 2, hàm sẽ không trả về số chuỗi kí tự được tách ra từ chuỗi gốc trong phần tử đầu tiên. Trong trường hợp bạn dùng tham số này, nếu muốn lấy số phần tử trong mảng, bạn phải dùng hàm UBound.

- Ví dụ : giả sử tôi có một chuỗi như sau

$s = "Viet,Nam,que,huong,toi"

Nếu tôi gọi lệnh *$arr = StringSplit($s, ",")*, thì biến $arr[0] sẽ là 5. $arr[1] = "Viet", $arr[2] = "Nam", $arr[3] = "que", $arr[4] = "huong", $arr[5] = "toi".

Nếu tôi gọi lệnh *$arr = StringSplit($s, ",N")*, thì biến $arr[0] sẽ là 6. $arr[1] = "Viet", $arr[2] = "", $arr[3] = "am", $arr[4] = "que", $arr[5] = "huong", $arr[6] = "toi". Chú ý : bạn thấy rằng phần tử $arr[2] = "" bởi vì khi gặp kí tự phân cách "N" nó sẽ tiến hành lấy chuỗi trước đó tuy nhiên trước nó cũng là kí tự phân cách là dấu "," (sau từ "Viet") do đó không có bất kì kí tự nào trả về.

Nếu tôi gọi lệnh *$arr = StringSplit($s, ",N", 1)*, thì biến $arr[0] sẽ là 2, $arr[1] = "Viet", $arr[2] = "am,que,huong,toi".

Bạn có thể dùng vòng lặp sau đây để tiến hành liệt kê các chuỗi tách trong $arr trên bằng cách gõ lênh như sau :

For $i = 1 to $arr[0]

MsgBox(0, "StringSplit", $arr[$i])

Next

Bạn có thể dùng vòng lặp trên nếu bạn không thiết lập tham số flag = 2 (vô hiệu hóa trả về số chuỗi con được tách trong phần tử đầu tiên của biến mảng chứa kết quả của hàm StringSplit).

- Nếu bạn cung cấp trong tham số del là "", thì mỗi phần tử trong mảng (bắt đầu từ vị trí index = 1) sẽ được gán tương ứng với mỗi kí tự trong chuỗi string, tính từ trái qua phải.

Hàm quản lý tiến trình

1. ProcessClose("process")

- Công dụng : đóng một tiến trình đang chạy

- process : là tên tiến trình cần đóng

Ví dụ : *ProcessClose("notepad.exe")*

2. ProcessExists("process")

- Công dụng : kiểm tra xem một tiến trình nào đó có đang chạy trong hệ thống hay không.

- process là tên tiến trình cần kiểm tra.

Ví dụ : *If ProcessExists("WINWORD.exe") Then*

MsgBox(0, "Example", "Microsoft Word is running.")

EndIf

3. ProcessGetStats(["process"], [type])

- Công dụng : lấy thông tin về bộ nhớ hoặc thông tin IO của một tiến trình nào đó.

- process : là tên tiến trình cần lấy thông tin. Nếu không có tham số này, hàm sẽ lấy thông tin của tiến trình hiện tại.

- type : tham số tùy chọn, mặc định là 0. Nếu là 0, hàm sẽ trả về một mảng có kích thước là 2, phần tử đầu tiên chứa thông tin bộ nhớ bị tiến trình đó chiếm giữ, phần tử thứ hai chứa thông tin kích thước bộ nhớ tối đa mà tiến trình đó có thể chiếm giữ (đơn vị tính byte). Nếu tham số type = 1, hàm sẽ trả về một mảng có 6 phần tử. Phần tử thứ nhất chứa thông tin số thao tác đọc được thực hiện, phần tử thứ hai chứa thông tin số thao tác ghi được thực hiện, phần tử thứ ba chứa thông tin số thao tác I/O được thực hiện, phần thử thứ tư chứa số byte được đọc, phần tử thứ năm chứa số thao tác ghi, phần tử thứ sáu chứa thông tin số byte được dùng trong quá trình ghi và đọc của tiến trình.

Ví dụ : *$test = ProcessGetStats("WINWORD.exe", 1)*

MsgBox(0, "ProcessGetStats", $test[0] & " " & $test[1])

4. ProcessSetPriority("process", priority)

- Công dụng : thiết lập chế độ ưu tiên của CPU cho một tiến trình.

- process : là tên tiến trình cần thiết lập chế độ ưu tiên, priority là mức ưu tiến mà bạn cần thiết lập. Tham số priority có thể có các giá trị như sau : 0 (Idle/Low), 1 (Below Normal), 2 (Normal), 3 (Above Normal), 4 (High), 5 (Realtime – Bạn nên cẩn thận với tùy chọn này. Rất có thể nó làm cho hệ thống của bạn không ổn định).

Ví dụ : *ProcessSetPriority("WINWORD.exe", 0)*

5. ProcessList(["name"])

- Công dụng : lấy thông tin của những tiến trình đang chạy trên hệ thống.

- name là tham số tùy chọn của tên tiến trình. Nếu bạn cung cấp tên tiến trình cho tham số này, hàm chỉ trả thông tin cho tiến trình trùng tên với tham số name được cho bởi người dùng.

- Hàm này nếu gọi thành công sẽ trả về một mảng hai chiều. Trong đó chiều thứ hai luôn có kích thước là hai. Bạn xem kết cấu mảng trả về như sau :

$array[0][0] = số tiến trình có trong hệ thống
$array[1][0] = tên của tiến trình thứ nhất
$array[1][1] = số ID của tiến trình thứ nhất
$array[2][0] = tên của tiến trình thứ hai
$array[2][1] = số ID của tiến trình thứ hai
...
$array[n][0] = tên của tiến trình thứ n
$array[n][1] = số ID của tiến trình thứ n.

Ví dụ :
```
$list = ProcessList()
    for $i = 1 to $list[0][0]
        msgbox(0, "Thong tin tien trinh",$list[$i][0] & " – "& $list[$i][1])
    next
```

6. ProcessWait("process", [timeout])

- Công dụng : tạm dừng một đoạn mã script AutoIt đang thực thi cho đến khi một tiến trình nào đó tồn tại chạy trong hệ thống.

- process : là tên tiến trình cần kiểm tra. timeout là thời gian tối đa mà hàm này có hiệu lực (tính bằng s), tức là sau thời gian này nếu tiến trình đó chưa có chạy trong hệ thống hay không thì đoạn mã sau lệnh ProcessWait vẫn được thực thi. Nếu bạn không cung cấp tham số này mặc định sẽ là chờ vô thời hạn (cho đến khi tiến trình process chạy mới bắt đầu chạy tiếp script).

Ví dụ :
```
ProcessWait("notepad.exe", 6)
    MsgBox(0, "Test", "Chay thu roi biet")
```

7. ProcessWaitClose("process", [timeout])

- Công dụng : tạm dừng một đoạn mã script AutoIt đang thực thi cho đến khi một tiến trình nào đó không tồn tại chạy trong hệ thống.

- process : là tên tiến trình cần kiểm tra. timeout là thời gian tối đa mà hàm này có hiệu lực (tính bằng s), tức là sau thời gian này nếu tiến trình đó có chạy trong hệ thống hay không thì đoạn mã sau lệnh ProcessWaitClose vẫn được thực thi. Nếu bạn không cung cấp tham số này mặc định sẽ là chờ vô thời hạn (chờ cho đến khi tiến trình process không còn tồn tại mới bắt đầu chạy tiếp script).

Ví dụ :
```
ProcessWaitClose("notepad.exe", 6)
    MsgBox(0, "Test", "Chay thu roi biet")
```

8. DllStructCreate("Struct")

- Công dụng : tạo dữ liệu có cấu trúc tương tự như struct trong C/C++.

- Struct là chuỗi chứa thông tin cấu trúc cần tạo. Bạn tuân theo mẫu như sau : "typevar1 var1;typevar2 var2; typevar3 var3;…;typevarn varn". Trong đó typevar là kiểu của biến, còn var là tên biến cần tạo cấu trúc. Bạn tham khảo kiểu dữ liệu của biến trong bảng sau :

Kiểu của biến	Chi tiết của kiểu biến
BYTE	8bit(1byte) kí tự không dấu
BOOLEAN	8bit(1byte) kí tự không dấu (có thể dùng True hoặc False)
CHAR	8bit(1byte) kí tự ASCII
WCHAR	16bit(2byte) kí tự UNICODE
short	16bit(2bytes) số nguyên có dấu
USHORT	16bit(2bytes) số nguyên không dấu
WORD	16bit(2bytes) số nguyên không dấu
int	32bit(4bytes) số nguyên có dấu
long	32bit(4bytes) số nguyên có dấu
BOOL	32bit(4bytes) số nguyên có dấu (bạn có thể dùng từ True hoặc False để minh họa cho đúng hoặc sai)
UINT	32bit(4bytes) số nguyên không dấu
ULONG	32bit(4bytes) số nguyên không dấu
DWORD	32bit(4bytes) số nguyên không dấu
INT64	64bit(8bytes) số nguyên có dấu
UINT64	64bit(8bytes) số nguyên không dấu
ptr	32 hoặc 64bit số nguyên không dấu (phụ thuộc vào phiên bản X86 hay X64 của AutoIt được dùng).
HWND	32bit(4bytes) số nguyên

HANDLE	32bit(4bytes) số nguyên
float	32bit(4bytes) số thực
double	64bit(8bytes) số thực
INT_PTR, LONG_PTR, LRESULT, LPARAM	32 hoặc 64bit số nguyên có dấu (phụ thuộc vào phiên bản X86 hay X64 của AutoIt được dùng)
UINT_PTR, ULONG_PTR, DWORD_PTR, WPARAM	32 hoặc 64bit số nguyên không dấu(phụ thuộc vào phiên bản X86 hay X64 của AutoIt được dùng)

Ví dụ : tạo cấu trúc trong C như sau

```
struct diachi
{
        int sonha;
        char pho[20];
        char thanhpho[15];
}
```

Đoạn mã lệnh tương ứng trong AutoIt sẽ là :

$str="int sonha;char pho[20];char thanhpho[15]"; các dấu ; ngăn cách các tên biến bạn phải để sát tên biến (bên trái) và kiểu dữ liệu biến (bên phải).

```
$a = DllStructCreate($str)
if @error Then
   MsgBox(0,"DllStructCreate","Loi trong khi tao struct");
   exit
endif
```

9. DllStructSetData(struct, element, value, [index])

- Công dụng : gán dữ liệu cho một biến trong cấu trúc struct

- struct là tên biến cấu trúc được trả về thông qua hàm DllStructCreate. element là tên biến trong struct mà bạn cần gán dữ liệu. value là giá trị cần gán. index là tham số tùy chọn, áp dụng cho một trường biến trong struct là biến mảng, là vị trí trong mảng cần ghi phần tử (tính từ trái qua phải trong mảng – bắt đầu từ 1). Bạn cũng nên chú ý là không được dùng tham số này cho một trường nào đó không phải là một mảng. Nếu tham số index không có hoặc là giá trị Default, hàm sẽ ghi vào biến mảng bắt đầu từ vị trí 1 (vị trí đầu tiên).

Ví dụ : *$str="int sonha;char pho[20];char thanhpho[15]*

```
$a = DllStructCreate($str)
if @error Then
    MsgBox(0,"DllStructCreate","Loi trong khi tao struct")
    exit
endif
DllStructSetData($a, "sonha", 15); sonha = 15
DllStructSetData($a, "pho", "Hang Rong"); pho = "Hang Rong"
DllStructSetData($a, "thanhpho", "Ha Noi"); thanhpho = "Ha Noi"
```

10. DllStructGetData(struct, element, [index])

- Công dụng : lấy dữ liệu của một biến trong struct.

- struct : struct là tên biến cấu trúc được trả về thông qua hàm DllStructCreate. element là tên biến cần lấy giá trị về, bạn có thể dùng số (bắt đầu từ 1) để thay cho tên biến. index là tham số tùy chọn, áp dụng cho một biến là biến mảng, là vị trí trong mảng cần lấy dữ liệu (tính từ trái qua phải trong mảng – bắt đầu từ 1). Nếu tham số index không có hoặc là giá trị Default, hàm sẽ lấy tất cả các phần tử trong mảng.

Ví dụ : *$str="int sonha;char pho[20];char thanhpho[15]"*

```
$a = DllStructCreate($str)
if @error Then
    MsgBox(0,"DllStructCreate","Loi trong khi tao struct");
    exit
endif
DllStructSetData($a, "sonha", 15)
DllStructSetData($a, "pho", "Hang Rong")
DllStructSetData($a, "thanhpho", "Ha Noi")
```

MsgBox(0, "DllStructGetData", DllStructGetData($a, "sonha") & @CRLF & DllStructGetData ($a, "pho") & @CRLF & DllStructGetData($a, "thanhpho")); bạn có thể thay thế đoạn mã Msgbox như sau :

MsgBox(0, "DllStructSetData", DllStructGetData($a , 1) & @CRLF & DllStructGetData($a, 2) & @CRLF&DllStructGetData($a, 3))

11. DllStructGetSize(Struct)

- Công dụng : lấy kích thước của struct (tính bằng bytes)

- struct : struct là tên biến cấu trúc được trả về thông qua hàm DllStructCreate.

Ví dụ : *$str="int sonha;char pho[20];char thanhpho[15]"*

```
$a = DllStructCreate($str)
if @error Then
    MsgBox(0,"DllStructCreate","Loi trong khi tao struct");
```

```
    exit
endif
MsgBox(0, "DllStructGetSize", DllStructGetSize($a))
```
; trả về 40

12. DllStructGetPtr(struct, [element])

- Công dụng : lấy thông tin con trỏ của struct hoặc của một biến nào đó trong struct.

- struct : struct là tên biến cấu trúc được trả về thông qua hàm DllStructCreate. element là tên biến cần lấy thông tin con trỏ, bạn có thể dùng số (bắt đầu từ 1) để thay cho tên biến. Nếu tham số element không cung cấp, hàm sẽ lấy thông tin pointer của struct.

Ví dụ :
```
$str="int sonha;char pho[20];char thanhpho[15]"
$a = DllStructCreate($str)
if @error Then
    MsgBox(0,"DllStructCreate","Loi trong khi tao struct")
    exit
endif
MsgBox(0, "DllStructGetPtr", DllStructGetPtr($a))
```

Macro chứa thông tin hệ thống

Macro	Mô tả
@CPUArch	Loại CPU trong máy tính của bạn. Trả về "X86" nếu như là CPU 32-bit, trả về "X64" nếu là CPU 64-bit.
@KBLayout	Trả về đoạn mã code tương ứng với kiểu ngôn ngữ bàn phím mà bạn đang sử dụng. Xem Code_Language trong bảng phía dưới để biết ý nghĩa các code.
@MUILang	Trả về code coi xem hệ điều hành có hỗ trợ chế độ đa ngôn ngữ hay không. Xem thêm bảng Code_Language.
@OSArch	Trả về kiến trúc máy tính mà hệ thống bạn đang sử dụng. Các gía trị có thể là "X86", "X64", "IA64".
@OSLang	Trả về code tương ứng với ngôn ngữ mà hệ điều hành bạn đang sử dụng. Xem thêm bảng Code_Language.
@OSType	Thông tin loại hệ điều hành mà bạn đang sử dụng. Trả về "WIN32_NT" cho Windows NT / 2000 / XP / 2003 / Vista / 2008 / Win7 / 2008R2.

@OSVersion	Trả về phiên bản hệ điều hành mà bạn đang sử dụng. "WIN_2008R2" – Windows Server 2008 R2, "WIN_7" – Windows 7, "WIN_2008" – Windows Server 2008, "WIN_VISTA" – Windows Vista, "WIN_2003" – Windows 2003, "WIN_XP" – Windows XP, "WIN_XPe", "WIN_2000" – Windows 2000.
@OSBuild	Trả về số build của hệ điều hành bạn đang sử dụng. Ví dụ, Windows 2003 Server trả về 3790.
@OSServicePack	Trả về thông tin số Service Pack được cài đặt trên hệ điều hành của bạn.
@ComputerName	Tên máy tính của bạn trong mạng.
@UserName	Tên tài khoản người dùng hiện đăng nhập hệ thống.
@IPAddress1	Địa chỉ IP của card mạng thứ nhất. Thông thường sẽ trả về 127.0.0.1 trên một vài máy tính.
@IPAddress2	Trả về địa chỉ IP của card mạng thứ hai trên máy tính của bạn. Trả về 0.0.0.0 nếu không có card mạng.
@IPAddress3	Trả về địa chỉ IP của card mạng thứ ba trên máy tính của bạn. Trả về 0.0.0.0 nếu không có card mạng.
@IPAddress4	Trả về địa chỉ IP của card mạng thứ tư trên máy tính của bạn. Trả về 0.0.0.0 nếu không có card mạng.
@DesktopHeight	Trả về độ dài chiều cao của màn hình desktop (pixel). (độ phân giải dọc).
@DesktopWidth	Trả về độ dài chiều ngang của màn hình desktop (pixel). (độ phân giải ngang).
@DesktopDepth	Độ sâu của màn hình desktop (bits per pixel).
@DesktopRefresh	Tần số làm tươi màn hình (tính bằng Hz).

Code_Language

Code	Quốc gia	Code	Quốc gia
0436	Afrikaans	1407	German_Liechtenstei

041c	Albanian	408	Greek
0401	Arabic_Saudi_Arabia	040d	Hebrew
0801	Arabic_Iraq	0439	Hindi
0c01	Arabic_Egypt	040e	Hungarian
1001	Arabic_Libya	040f	Icelandic
1401	Arabic_Algeria	0421	Indonesian
1801	Arabic_Morocco	0410	Italian_Standard
1c01	Arabic_Tunisia	0810	Italian_Swiss
2001	Arabic_Oman	0411	Japanese
2401	Arabic_Yemen	043f	Kazakh
2801	Arabic_Syria	0457	Konkani
2c01	Arabic_Jordan	0412	Korean
3001	Arabic_Lebanon	0426	Latvian
3401	Arabic_Kuwait	0427	Lithuanian
3801	Arabic_UAE	042f	Macedonian
3c01	Arabic_Bahrain	043e	Malay_Malaysia
4001	Arabic_Qatar	083e	Malay_Brunei_Darussalam
042b	Armenian	044e	Marathi
042c	Azeri_Latin	0414	Norwegian_Bokmal
082c	Azeri_Cyrillic	0814	Norwegian_Nynorsk
042d	Basque	0415	Polish
0423	Belarusian	0416	Portuguese_Brazilian

0402	Bulgarian	0816	Portuguese_Standard
0403	Catalan	0418	Romanian
0404	Chinese_Taiwan	0419	Russian
0804	Chinese_PRC	044f	Sanskrit
0c04	Chinese_Hong_Kong	081a	Serbian_Latin
1004	Chinese_Singapore	0c1a	Serbian_Cyrillic
1404	Chinese_Macau	041b	Slovak
041a	Croatian	0424	Slovenian
0405	Czech	040a	Spanish_Traditional_Sort
0406	Danish	080a	Spanish_Mexican
0413	Dutch_Standard	0c0a	Spanish_Modern_Sort
0813	Dutch_Belgian	100a	Spanish_Guatemala
0409	English_United_States	140a	Spanish_Costa_Rica
0809	English_United_Kingdom	180a	Spanish_Panama
0c09	English_Australian	1c0a	Spanish_Dominican_Republic
1009	English_Canadian	200a	Spanish_Venezuela
1409	English_New_Zealand	240a	Spanish_Colombia
1809	English_Irish	280a	Spanish_Peru
1c09	English_South_Africa	2c0a	Spanish_Argentina
2009	English_Jamaica	300a	Spanish_Ecuador
2409	English_Caribbean	340a	Spanish_Chile
2809	English_Belize	380a	Spanish_Uruguay

2c09	English_Trinidad	3c0a	Spanish_Paraguay
3009	English_Zimbabwe	400a	Spanish_Bolivia
3409	English_Philippines	440a	Spanish_El_Salvador
0425	Estonian	480a	Spanish_Honduras
0438	Faeroese	4c0a	Spanish_Nicaragua
0429	Farsi	500a	Spanish_Puerto_Rico
040b	Finnish	0441	Swahili
040c	French_Standard	041d	Swedish
080c	French_Belgian	081d	Swedish_Finland
0c0c	French_Canadian	0449	Tamil
100c	French_Swiss	0444	Tatar
140c	French_Luxembourg	041e	Thai
180c	French_Monaco	041f	Turkish
0437	Georgian	0422	Ukrainian
0407	German_Standard	0420	Urdu
0807	German_Swiss	0443	Uzbek_Latin
0c07	German_Austrian	0843	Uzbek_Cyrillic
1007	German_Luxembourg	042a	Vietnamese
1407	German_Liechtenstei		

Các hàm liên quan đến GUI (Graphic User Interface)

1. GUICreate("title", [width], [height], [left], [top], [style], [exstyle], [parent])

- Công dụng : tạo một cửa sổ Windows trống (bạn cần phải tạo cửa sổ này trước khi tạo các control khác như Label, Textbox, Listbox,… Tuy nhiên hàm này chỉ mới có công dụng tạo cửa sổ thôi chứ nó chưa xuất hiện trên màn hình khi chạy. Để làm cho nó xuất hiện bạn tham khảo thêm hàm *GUISetState*.

- Hàm này nếu gọi thành công sẽ trả về handle của cửa sổ được tạo trong hàm. Nói một cách dễ hiểu handle là tên đại diện của cửa sổ được tạo ra.

- Bảng thông số hàm

title	Tiêu đề của cửa sổ cần tạo (tiêu đề này sẽ xuất hiện trên thanh tiêu đề của cửa sổ).
width	Độ dài chiều ngang của cửa sổ cần tạo (tính bằng pixel).
height	Độ dài chiều cao của cửa sổ cần tạo (tính bằng pixel).
left	Vị trí bên trái của cửa sổ (tọa độ X), mặc định là -1 – canh giữa màn hình. Nếu tham số này được định nghĩa, bạn cũng phải cung cấp thêm tham số top.
top	Tọa độ đỉnh trên của cửa sổ (tọa độ Y). Mặc định là -1 – canh giữa màn hình.
style	Thiết lập kiểu dáng (style) của cửa sổ. Tham số mặc định là -1 sẽ kết hợp các style $WS_MINIMIZEBOX, $WS_CAPTION, $WS_POPUP, $WS_SYSMENU. Bạn có thể tham khảo danh sách kiểu style trong bảng Style dưới đây.
exstyle	Thiết lập kiểu dáng style mở rộng của cửa sổ. Mặc định là -1. Xem bảng Exstyle để biết thêm các kiểu style mở rộng.
parent	Biến handle của cửa sổ tạo trước đó – cửa sổ được tạo trong hàm này sẽ trở thành cửa sổ con của nó.

Bảng Style (tham khảo từ AutoIt Help, bạn có thể tham khảo thêm các tham số khác trong bảng ở tài liệu AutoIt Help).

Tham số style	**Mô tả**
Để dùng được các style sau đây, bạn cần phải thêm câu lệnh #include<WindowsConstants.au3> ngay tại phần đầu chương trình. Để kết hợp được các style với nhau bạn hãy dùng BitOr(name_style_1, name_style_2,… name_style_n).	
$WS_BORDER	Tạo một cửa sổ có đường biên đóng khung xung quanh.
$WS_POPUP	Tạo một cửa sổ dạng pop-up. Style này không thể dùng chung với style WS_CHILD.

$WS_CAPTION	Tạo ra một cửa sổ có thanh tiêu đề (bao gồm style WS_BORDER).
$WS_CLIPCHILDREN	Tham số này sử dụng khi bạn tạo một cửa sổ, sau đó sẽ tạo ra cửa sổ con chứa trong đó (áp dụng cho các cửa sổ cha).
$WS_DISABLED	Tạo một cửa sổ bị vô hiệu hóa.
$WS_DLGFRAME	Tạo một cửa sổ có đường biên đóng bên ngoài, tương tự như kiểu dáng của các hộp thoại Dialog Box.
$WS_HSCROLL	Tạo một cửa sổ có thanh trượt ngang phía dưới cửa sổ.
$WS_MAXIMIZE	Tạo một cửa sổ ban đầu phóng lớn toàn màn hình.
$WS_MAXIMIZEBOX	Tạo một cửa sổ có nút Maximize. Không kết hợp được với style WS_EX_CONTEXTHELP.
$WS_MINIMIZE	Tạo một cửa sổ ban đầu được thu nhỏ trên thanh taskbar.
$WS_MINIMIZEBOX	Tạo một cửa sổ có nút Minimize. Không kết hợp được với style WS_EX_CONTEXTHELP.
$WS_OVERLAPPED	Tạo ra một cửa sổ theo kiểu chồng chéo lên nhau. Loại cửa sổ này có thanh tiêu đề và được đóng khung. Tương tự như style WS_TILED.
$WS_OVERLAPPEDWINDOW	Tạo một cửa sổ chồng chéo với các tham số như WS_OVERLAPPED, WS_CAPTION, WS_SYSMENU, WS_THICKFRAME,WS_MINIMIZEBOX, và style WS_MAXIMIZEBOX. Tương tự như style WS_TILEDWINDOW.

$WS_POPUPWINDOW	Tạo cửa sổ popup với các tham số WS_BORDER, WS_POPUP, và style WS_SYSMENU. Tham số style WS_CAPTION và style WS_POPUPWINDOW cần phải kết hợp với nhau để làm cho thanh thực đơn của cửa sổ có thể khả dụng.
$WS_SIZEBOX	Tạo một cửa sổ có thể thay đổi kích thước. Tương tự như style WS_THICKFRAME.
$WS_SYSMENU	Tạo một cửa sổ có thanh thực đơn trên thanh tiêu đề của nó.
$WS_THICKFRAME	Tạo một cửa sổ có thể thay đổi kích thước. Tương tự như style WS_SIZEBOX style
$WS_VSCROLL	Tạo một cửa sổ có thanh trượt dọc.
$WS_VISIBLE	Tạo cửa sỏ ban đầu có thể sử dụng được (không bị vô hiệu hóa).
$WS_CHILD	Tạo một cửa sổ con. Style của cửa sổ này không thể có thanh thực đơn. Style này không được dùng với style WS_POPUP.
$WS_GROUP	Tạo một cửa sổ để tập hợp các control thành một nhóm. Cửa sổ này sẽ chứa control đầu tiên và các control được tạo sau nó.
$WS_TABSTOP	Tạo một cửa sổ có thể dùng phím Tab để di chuyển từ control này sang control khác.
$DS_MODALFRAME	Tạo ra một khung hộp thoại kết hợp thanh tiêu đề và menu của cửa sổ bằng cách xác định thêm tham số WS_CAPTION và WS_SYSMENU.

$DS_SETFOREGROUND	Kiểu dáng này của cửa sổ nhằm thu hút sự chú ý của người dùng đến cửa sổ này (cửa sổ này không thể thu nhỏ trên thanh taskbar).
$DS_CONTEXTHELP	Kiểu dáng của cửa sổ này sẽ có một dấm chấm hỏi được đặt trên thanh tiêu đề. Style này không thể dùng với style WS_MAXIMIZEBOX hoặc WS_MINIMIZEBOX. Style này tương tự như style WS_EX_CONTEXTHELP.

Bảng tham số Exstyle (tham khảo từ AutoIt Help).

Extended Style	Mô tả
Để dùng được các tham số sau đây thì bạn cần phải thêm chỉ thị #include <WindowsConstants.au3> ngay tại phần đầu chương trình.	
$WS_EX_ACCEPTFILES	Cho phép các control edit hoặc input được tạo trong cửa sổ GUI có thể xác định tên tập tin thông qua cách kéo và thả (drag and drop). Các control phải được gán trạng thái $GUI_DROPACCEPTED được thiết lập bởi hàm GUICtrlSetState. Đối với các control khác, thông tin drag & drag có thể lấy về với @GUI_DRAGID, @GUI_DRAGFILE, @GUIDROPID.
$WS_EX_APPWINDOW	Bắt buộc một cửa sổ cấp cao nhất luôn ở trên thanh taskbar khi nó xuất hiện.
$WS_EX_CLIENTEDGE	Cửa sổ sẽ được đóng với khung bị chìm xuống.
$WS_EX_CONTEXTHELP	Bao gồm một dấu chấm hỏi trên thanh tiêu đề của cửa sổ. Không thể dùng với WS_MAXIMIZEBOX hoặc WS_MINIMIZEBOX.
$WS_EX_DLGMODALFRAME	Tạo một cửa sổ được đóng khung đôi ; bạn có thể tùy chọn cửa sổ được tạo với thanh tiêu đề bằng cách chỉ định WS_CAPTION trong tham số style.
$WS_EX_MDICHILD	Tạo một cửa sổ con nằm bên trong cửa sổ bố mẹ (mô phỏng không phải MDI thực sự).
$WS_EX_OVERLAPPEDWINDOW	Kết hợp style WS_EX_CLIENTEDGE và WS_EX_WINDOWEDGE styles.

$WS_EX_STATICEDGE	Tạo một cửa sổ được đóng khung ba chiều dự định sẽ được dùng cho các mục mà không chấp nhận cho người dùng nhập vào.
$WS_EX_TOPMOST	Cửa sổ được tạo bởi style này sẽ luôn nằm trên các cửa sổ khác, kể cả khi chúng không được kích hoạt.
$WS_EX_TRANSPARENT	Cửa sổ sẽ xuất hiện trong suốt bởi vì các bit nằm dưới cửa sổ có liên quan với nó đã được vẽ lại.
$WS_EX_TOOLWINDOW	Tạo ra một cửa sổ chứa thanh công cụ, một cửa sổ dự định được dùng như thanh công cụ trôi. Một cửa sổ chứa thanh công cụ sẽ có thanh tiêu đề ngắn hơn một thanh tiêu đề bình thường và tiêu đề của cửa sổ này sẽ dùng một font chữ nhỏ hơn. Một cửa sổ chứa thanh công cụ sẽ không được xuất hiện trên thanh tác vụ hoặc trong hộp thoại xuất hiện khi người dùng nhấn phím ALT – TAB. Nếu cửa sổ chứa công cụ có hệ thống menu, biểu tượng của nó không được hiển thị trên thanh tiêu đề. Tuy nhiên, bạn có thể hiển thị menu hệ thống bằng cách nhấn tổ hợp phím ALT – SPACE.
$WS_EX_WINDOWEDGE	Tạo một cửa sổ được đóng khung với viền nổi.
$WS_EX_LAYERED	Tạo ra một cửa sổ xếp thành lớp. Lưu ý là tham số này không được phép sử dụng cho các cửa sổ con.

Ví dụ : *#include <WindowsConstants.au3>*

$gui = GUICreate("Background", 400, 100, 50, 60, $WS_SIZEBOX)

Hàm xử lý cửa sổ được tạo

1. GUIDelete([winhandle])

- Công dụng : xóa một cửa sổ GUI và tất cả control ở trong đó.

- winhandle là biến "handle" được trả về trong hàm GUICreate. Nếu bạn không cung cấp tham số này, hàm sẽ tiến hành xóa GUI được dùng trước đó.

Ví dụ : *#include <WindowsConstants.au3>*

$gui = GUICreate("Background", 400, 100, 50, 60, $WS_SIZEBOX)

GUISetState(@SW_SHOW); hiện cửa sổ

Sleep(4000)

GUIDelete($gui)

MsgBox(0, "GUIDelete", "Da xoa xong cua so")

2. GUIGetCurosInfo([winhandle])

- Công dụng : lấy thông tin vị trí con trỏ chuột trên cửa sổ hiện tại.

- winhandle là biến "handle" của cửa sổ được trả về trong hàm GUICreate. Nếu bạn

không cung cấp tham số này, hàm sẽ tiến hành lấy cửa sổ hiện hành để lấy thông tin con trỏ chuột.

- Hàm này nếu gọi thành công sẽ trả về một mảng có năm phần tử. Phần tử thứ nhất chứa tọa độ X (chiều ngang), phần tử thứ hai chứa tọa độ Y (chiều dọc). Phần tử thứ ba chứa thông tin nút chính của chuột có được nhấn hay không (thường là nút bên trái chuột), trả về 1 nếu được nhấn, trả về 0 nếu không được nhấn. Phần tử thứ tư chứa thông tin nút thứ hai của chuột có được nhấn hay không (thường là nút bên phải chuột), trả về 1 nếu được nhấn, trả về 0 nếu không được nhấn. Phần tử thứ năm chứa ID của control mà con chuột di chuyển trên nó (trả về 0 nếu không nếu không có control nào mà chuột di chuyển trên nó).

- Tọa độ con trỏ chuột được tính bên trong cửa sổ GUI.

- Nếu tham số winhandle được cho, thì cửa sổ có biến handle đó sẽ trở thành cửa sổ hiện hành.

- Các control như ListViewItem hoặc TreeView thì sẽ không trả về ID của control đó, chỉ trả về số ID của ListView hay TreeView mà thôi.

Ví dụ : *#include <WindowsConstants.au3>*

```
$gui = GUICreate("Background", 800, 600, -1, -1, $WS_SIZEBOX)
GUISetState(@SW_SHOW)
Sleep(4000)
$tam = GUIGetCursorInfo($gui)
MsgBox(0, "GUIDelete", "Toa do X : "&$tam[0]&" Toa do Y : "&$tam[1])
```

3. GUIGetMsg([advanced])

- Công dụng : trả về thông tin sự kiện nào xảy ra trên GUI.

- advanced : là tham số quy định cách trả thông tin về của hàm. Nếu tham số này là 0, hàm sẽ trả về một sự kiện nào đó mà thôi. Nếu tham số này là 1, hàm sẽ trả về một mảng có năm phần tử chứa thông tin sự kiện và các thông tin mở rộng. Phần tử thứ nhất chứa Event ID hoặc Control ID hoặc trả về 0 nếu không có sự kiện nào xảy ra. Phần tử thứ hai chứa thông tin "handle" của cửa sổ xảy ra sự kiện. Phần tử thứ ba chứa thông tin "handle" của control xảy ra sự kiện. Phần tử thứ tư chứa thông tin tọa độ X của con trỏ chuột hiện hành. Phần tử thứ năm chứa thông tin tọa độ Y của con trỏ chuột hiện hành.

Bảng Event ID tương ứng các sự kiện

Event ID	Sự kiện xảy ra
Nếu muốn dùng các Event ID này để kiểm tra xem sự kiện nào đó được xảy ra trên hộp thoại thì bạn phải thêm chỉ thị #include <GUIConstantsEx.au3> ngay tại phần đầu chương trình.	
0	Không có sự kiện nào xảy ra
$GUI_EVENT_CLOSE	Hộp thoại bị đóng
$GUI_EVENT_MINIMIZE	Hộp thoại sẽ bị thu nhỏ trên thanh taskbar
$GUI_EVENT_RESTORE	Hộp thoại xuất hiện trở lại do nhấn biểu tượng của nó trên thanh taskbar.

$GUI_EVENT_MAXIMIZE	Hộp thoại được phóng lớn toàn màn hình.
$GUI_EVENT_MOUSEMOVE	Con trỏ chuột đang di chuyển
$GUI_EVENT_PRIMARYDOWN	Nút chính trên con chuột được nhấn (thường là nút trái chuột)
$GUI_EVENT_PRIMARYUP	Nút chính trên con chuột được nhả ra (thường là nút trái chuột).
$GUI_EVENT_SECONDARYDOWN	Nút thứ hai trên con chuột được nhấn (thường là nút phải chuột).
$GUI_EVENT_SECONDARYUP	Nút thứ hai trên con chuột được nhả ra (thường là nút phải chuột).
$GUI_EVENT_RESIZED	Hộp thoại bị thay đổi kích thước
$GUI_EVENT_DROPPED	Sự kiện kéo thả trên cửa sổ kết thúc. @GUI_DRAGID, @GUI_DRAGFILE và @GUI_DROPID sẽ được dùng để lấy thông tin về số ID của control hoặc tên tập tin đối với các control có liên quan.

Ví dụ :
```
#include <WindowsConstants.au3>
#include <GUIConstantsEx.au3>
$gui = GUICreate("Background", 800, 600, -1, -1, $WS_SIZEBOX)
GUISetState(@SW_SHOW)
 While 1
    $msg = GUIGetMsg()
    Select
       Case $msg = $GUI_EVENT_CLOSE
          MsgBox(0, "GUIGetMsg", "Hop thoai bi dong")
          Exit
       Case $msg = $GUI_EVENT_MINIMIZE
          MsgBox(0, "GUIGetMsg", "Hop thoai duoc thu nho tren thanh taskbar")
       Case $msg = $GUI_EVENT_MAXIMIZE
          MsgBox(0, "GUIGetMsg", "Hop thoai duoc phong lon toan man hinh")
    EndSelect
  Wend
```

4. GUIGetStyle([winhandle])

- Công dụng : trả về phong cách style của cửa sổ được tạo.

- winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó.

- Hàm này nếu gọi thành công sẽ trả về một mảng có hai phần tử. Phần tử thứ nhất chứa thông tin style của cửa sổ, phần tử thứ hai chứa thông tin style mở rộng (exstyle) của cửa sổ.

Ví dụ : *#include <WindowsConstants.au3>*

```
$gui = GUICreate("Background", 800, 600, -1, -1, $WS_SIZEBOX)
GUISetState(@SW_SHOW)
$tam = GUIGetStyle($gui)
MsgBox(0, "GUIGetStyle", "Style : " & Hex($tam[0]) & " Exstyle : "& Hex($tam[1]))
```

5. GUIStartGroup([winhandle])

- Công dụng : các điều khiển control sau hàm này sẽ được nhóm lại với nhau.

- winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó.

- Hàm này thường được sử dụng với khi làm việc với các điều khiển control radio button. Khi bạn click một nút radio thì các radio button còn lại trong nhóm sẽ được reset.

6. GUISwitch(winhandle, [controlID])

- Công dụng : gán cửa sổ mới thành cửa sổ hiện hành. Tuy nhiên không có nghĩa là hàm này sẽ làm cho cửa sổ đó kích hoạt.

- winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó. controlID là số ID của control tabitem được chọn.

7. GUISetAccelerators(accelerators, [winhandle])

- Công dụng : thiết lập tổ hợp phím nóng cho các control trong một cửa sổ GUI.

- accelerators : là một mảng hai chiều, trong đó chiều thứ hai luôn có kích thước là hai. Mảng sẽ có dạng như sau :

$array[0][0] = phím nóng cho control thứ nhất.

$array[0][1] = số ID của control thứ nhất cần gán phím nóng.

$array[1][0] = phím nóng cho control thứ hai.

$array[1][1] = số ID của control thứ hai cần gán phím nóng.

...
$array[n][0] = phím nóng cho control thứ n.

$array[n][1] = số ID control thứ n cần gán phím nóng.

Ví dụ : *#include <GUIConstantsEx.au3>*

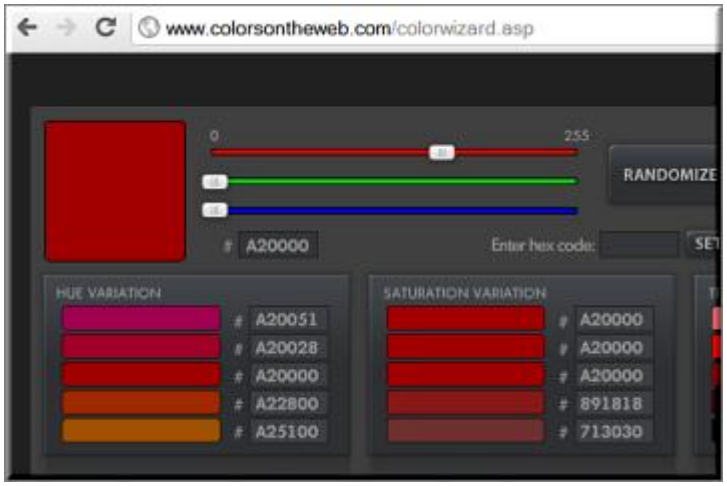
```
GUICreate("Custom Msgbox", 210, 80)
GUICtrlCreateLabel("Click”, 10, 10)
```

```
$YesID = GUICtrlCreateButton("Yes", 10, 50, 50, 20)
$NoID = GUICtrlCreateButton("No", 80, 50, 50, 20)
$ExitID = GUICtrlCreateButton("Exit", 150, 50, 50, 20)
; Gán phím nóng cho button Yes (Ctrl + y), button No (Ctrl + n).
Dim $AccelKeys[2][2]=[["^y", $YesID], ["^n", $NoID]]
GUISetAccelerators($AccelKeys)
```

8. GUISetBkColor(background, [winhandle])

- Công dụng : thiết lập màu nền cho một cửa sổ.

- background là mã màu cần thiết lập màu nền cho cửa sổ (thêm tiền tố 0x trước mã màu cần thiết lập). winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó. Để lấy mã màu cho một màu nào đó, bạn có thể vào trang web http://www.colorsontheweb.com/colorwizard.asp. Sau đó bạn hãy chọn một màu ưa thích và chép lấy mã màu tương ứng của nó.



Ví dụ : *GUICreate("Windows Background")* ; tạo một hộp thoại sẽ xuất hiện ở giữa.

GUISetBkColor(0xFF0808) ; thay đổi màu nền của cửa sổ.

GUISetState() ; hiện cửa sổ

Sleep(4000)

9. GUISetCoord(left, top, [width], [height], [winhandle])

- Công dụng : thiết lập vị trí tuyệt đối cho các control được tạo ra tiếp theo sau lệnh này.

- left : vị trí tuyệt đối bên trái của control (tọa độ X), top : vị trí tuyệt đối đỉnh trên của control (tọa độ Y). width : độ dài chiều ngang của control, height : độ dài chiều cao của control. winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó.

Ví dụ : *#include <GUIConstantsEx.au3>*

GUICreate("My GUI Set Coord", 200, 100)

GUISetCoord(500, 60)

```
GUICtrlCreateButton("OK #3", -1, -1)
GUICtrlCreateButton("Cancel #4", 10, -1)
GUISetState()
Sleep(2000)
```

10. GUISetFont(size, [weight], [attrib], [fontname], [winhandle], [quality])

- Công dụng : thiết lập font chữ cho cửa sổ GUI

- size : là kích thước font chữ (mặc định là 8.5). weight : thiết lập độ đậm cho font chữ. attrib : thiết lập thuộc tính cho font chữ như nghiêng (2), gạch dưới (4), gạch ngang từ (8). Bạn có thể kết hợp các thuộc tính font chữ bằng cách gõ số tổng cộng cho từng thuộc tính. Ví dụ tham số attrib = 6 sẽ gán thuộc tính vừa in nghiêng vừa gạch dưới. fontname là tên font cần sử dụng, nếu tham số này là "" hoặc tên font trong tham số này không tìm thấy, hàm sẽ lấy font mặc định của hệ điều hành. winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó. quality là chất lượng của font chữ, mặc định là PROOF_QUALITY = 2.

Ví dụ : *GUICreate("Student")* ; tạo một hộp thoại sẽ hiển thị ở giữa màn hình.

```
$font = "Verdana"
GUISetFont(15, 400, 8, $font)
GUICtrlCreateLabel("Mr. Trinh - DH10TH", 10, 20)
GUISetState()
Sleep(3000)
```

11. GUISetHelp(helpfile, [winhandle])

- Công dụng : chạy một tập tin khi bạn nhấn phím F1.

- helpfile là đường dẫn đến tập tin cần chạy. winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó.

Ví dụ : *GUICreate("Student")*

```
$font = "Verdana"
GUISetFont(15, 400, 4, $font)
GUISetState()
GUISetHelp("C:\Program Files\CCleaner\CCleaner.exe")
Sleep(8000)
```

12. GUISetIcon(iconfile, [iconid], [winhandle])

- Công dụng : thiết lập icon cho cửa sổ.

- iconfile : là đường dẫn đến tập tin cần làm icon. iconid là số ID của icon cần dùng trong tập tin iconfile, mặc định là -1 (tham số này thường có khi sử dụng các tập tin icon có dạng *.dll). winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó.

Ví dụ : *GUICreate("Student")*

$font = "Verdana"

GUISetFont(15, 400, 4, $font)

GUISetState()

GUISetIcon(@SystemDir &"\shell32.dll", 4); lần lược thay số iconid để xem các icon khác trong chương trình này.

Sleep(3000)

13. GuiSetOnEvent(specialID, "function", [winhandle])

- Công dụng : gọi một hàm nào đó khi một sự kiện của cửa sổ xảy ra. Để sử dụng hàm này, bạn thêm tham chỉ thị #include <GUIConstantsEx.au3> ngay tại phần đầu chương trình và thêm chỉ thị Opt("GUIOnEventMode", 1).

- specialID là mã sự kiện của cửa sổ, xem bảng bên dưới. function là tên hàm cần gọi khi sự kiện của cửa sổ xảy ra. winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó.

Bảng sự kiện

$GUI_EVENT_CLOSE	Hộp thoại bị đóng
$GUI_EVENT_MINIMIZE	Hộp thoại sẽ bị thu nhỏ trên thanh taskbar
$GUI_EVENT_RESTORE	Hộp thoại xuất hiện trở lại do nhấn biểu tượng của nó trên thanh taskbar.
$GUI_EVENT_MAXIMIZE	Hộp thoại được phóng lớn toàn màn hình.
$GUI_EVENT_MOUSEMOVE	Con trỏ chuột đang di chuyển
$GUI_EVENT_PRIMARYDOWN	Nút chính trên con chuột được nhấn (thường là nút trái chuột)
$GUI_EVENT_PRIMARYUP	Nút chính trên con chuột được nhả ra (thường là nút trái chuột).
$GUI_EVENT_SECONDARYDOWN	Nút thứ hai trên con chuột được nhấn (thường là nút phải chuột).
$GUI_EVENT_SECONDARYUP	Nút thứ hai trên con chuột được nhả ra (thường là nút phải chuột).
$GUI_EVENT_RESIZED	Hộp thoại bị thay đổi kích thước
$GUI_EVENT_DROPPED	Sự kiện kéo thả trên cửa sổ kết thúc. @GUI_DRAGID, @GUI_DRAGFILE và @GUI_DROPID sẽ được dùng để lấy thông tin về số ID của control hoặc tên tập tin đối với các control có liên quan.

Ví dụ : *#include <GUIConstantsEx.au3>*

Opt("GUIOnEventMode", 1)

GUICreate("Student")

GUISetState()

GUISetOnEvent($GUI_EVENT_CLOSE, "Close")

Sleep(3000)

Func Close()

MsgBox(4, "Warning", "Ban co muon thoat khoi chuong trinh khong ?")

EndFunc

14. GUISetState([flag], [winhandle])

- Công dụng : thiết lập trạng thái cho cửa sổ được tạo trước đó.

- flag : tham số quyết định trạng thái của cửa sổ (xem bảng tham số flag dưới đây). winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó.

Bảng tham số flag (tham khảo từ AutoIt Help)

@SW_HIDE	Ẩn cửa sổ GUI
@SW_SHOW	Hiện cửa sổ đã bị ẩn trước đó.
@SW_MINIMIZE	Thu nhỏ cửa sổ trên thanh taskbar
@SW_MAXIMIZE	Phóng lớn cửa sổ toàn màn hình
@SW_RESTORE	Phục hồi cửa sổ đã bị thu nhỏ trên thanh taskbar.
@SW_DISABLE	Vô hiệu hóa cửa sổ
@SW_ENABLE	Phục hồi cửa sổ đã bị vô hiệu hóa trước đó
@SW_LOCK	Khóa cửa sổ không cho vẽ
@SW_UNLOCK	Mở khóa cửa sổ cho phép vẽ.

Ví dụ : *GUICreate("Student")*

GUISetState(@SW_SHOW)

GUISetState(@SW_DISABLE)

Sleep(3000)

15. GUISetStyle(style, [exstyle], [winhandle])

- Công dụng : thiết lập kiểu style cho cửa sổ GUI. Thêm chỉ thị #include <WindowsConstants.au3> để sử dụng các đối số của hàm.

- style là kiểu mà bạn muốn gán cho cửa sổ. exstyle là style mở rộng cho cửa sổ (mặc

định là -1). winhandle là biến handle của cửa sổ được tạo ra bởi hàm GUICreate. Nếu bạn không cung cấp tham số này, mặc định sẽ là cửa sổ được dùng trước đó.

- Hai tham số style và exstyle bạn có thể xem các giá trị của hai tham số này trong hàm GUICreate.

Ví dụ : *#include <WindowsConstants.au3>*

GUICreate("Student")

GUISetStyle($WS_POPUP)

GUISetState()

Sleep(3000)

Hàm tạo các control

1. GUICtrlCreateLabel("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một label tĩnh trong một cửa sổ.

- text : văn bản của label sẽ hiển thị. left : vị trí bên trái của label (tọa độ X). top : vị trí đỉnh trên của label (tọa độ Y). width : là độ rộng của label, mặc định sẽ là vừa khít với nội dung text. height : là chiều cao của label, mặc định sẽ là vừa với chiều cao của text. style : kiểu dáng của label (xem bảng dưới), exstyle : style mở rộng của label (xem bảng trong hàm GUICreate).

- Hàm này nếu gọi thành công sẽ trả về số ID của label đó.

Bảng style cho control label (tham khảo AutoIt Help)

Style	Mô tả
Để sử dụng các style sau đây, bạn phải thêm chỉ thị #include <StaticConstants.au3> ngay tại phần đầu chương trình.	
Mặc định	$GUI_SS_DEFAULT_LABEL, $GUI_SS_DEFAULT_ICON, $GUI_SS_DEFAULT_PIC xem thêm hàm GUICtrlCreateLabel, GUICtrlCreateIcon, GUICtrlCreatePic.
$SS_BLACKFRAME	Label sẽ có một khung hình chữ nhật bao xung quanh, màu đen sẽ là màu mặc định của khung này.
$SS_BLACKRECT	Label sẽ có hình dáng chữ nhật với nền màu đen.
$SS_CENTER	Label chỉ xuất hiện text và text sẽ canh giữa ở khung chứa label. Chữ trong văn bản sẽ được di chuyển xuống dưới dòng mới khi không đủ chứa nội dung của một hàng và cũng sẽ được canh giữa.

$SS_CENTERIMAGE	Điểm giữa của một control tĩnh với phong cách style SS_BITMAP sẽ vẫn cố định khi bạn thay đổi kích thước của control. Bốn bên của control sẽ được điều chỉnh cho phù hợp với một ảnh bitmap. Nếu bitmap nhỏ hơn so với control. Phần còn thừa lại sẽ có màu trùng với màu của điểm ảnh (pixel) ở góc trên bên trái của bitmap. Nó có thể được sử dụng với một control tĩnh chỉ có một dòng văn bản.
$SS_ETCHEDFRAME	Control sẽ được đóng khung bằng cách sử dụng style EDGE_ETCHED.
$SS_ETCHEDHORZ	Cạnh trên hoặc cạnh dưới của control sẽ dùng style EDGE_ETCHED (dùng để gây chú ý).
$SS_ETCHEDVERT	Cạnh trái hoặc cạnh phải của control sẽ dùng style EDGE_ETCHED (dùng để gây chú ý).
$SS_GRAYFRAME	Khung label sẽ được đóng khung và màu của viền khung sẽ trùng màu với màu nền của cửa sổ hiện tại. Màu xám sẽ là màu mặc định cho màu viền khung.
$SS_GRAYRECT	Label sẽ có dạng hình chữ nhật và có màu nền trùng với màu nền của cửa sổ hiện tại. Màu xám sẽ là màu nền mặc định của label.
$SS_LEFT	Khung label sẽ có dạng hình chữ nhật đơn giản và nội dung văn bản trong label đó sẽ được canh trái. Văn bản sẽ được định dạng trước khi nó được hiển thị. Các từ cuối trong cùng một dòng nếu quá chiều dài của label sẽ được tự động di chuyển xuống dòng mới. Những từ nào quá chiều dài của control cũng sẽ được cắt ngắn đi.
$SS_LEFTNOWORDWRAP	Khung label sẽ có dạng hình chữ nhật và nội dung văn bản trong label đó sẽ được canh trái. Tuy nhiên nội dung văn bản nếu dài quá chiều dài của label sẽ được che bớt đi cho phù hợp với chiều dài của label.
$SS_NOPREFIX	Ngăn chặn dấu & trong control. Một ứng dụng có thể kết hợp style SS_NOPREFIX với các style khác bằng cách dùng toán tử OR(|). Điều này thật sự hữu ích khi tên tập tin hoặc các chuỗi khác có chứa kí hiệu & phải được hiển thị trong một control.

$SS_NOTIFY	Gửi đến cửa sổ cha của cửa sổ hiện thời thông điệp STN_CLICKED khi người dùng nhấp chuột vào control.
$SS_RIGHT	Khung label sẽ có hình dạng chữ nhật và nội dung văn bản bên trong nó sẽ được canh phải.
$SS_RIGHTJUST	Chỉ định rằng góc dưới bên phải của một control sẽ mang style SS_BITMAP hoặc SS_ICON và vẫn sẽ cố định khi control bị thay đổi kích cỡ. Chỉ có bên trên và bên trái được điều chỉnh để thích ứng với một bitmap mới hoặc một icon.
$SS_SIMPLE	Khung label sẽ là một hình chữ nhật đơn giản và văn bản nằm bên trong nó sẽ được canh lề trái. Văn bản trong label không thể thay đổi bằng bất cứ cách nào.
$SS_SUNKEN	Khung label sẽ được đóng khung thành hình chữ nhật tuy nhiên sẽ bị in chìm đi.
$SS_WHITEFRAME	Label sẽ có dạng một khung hình chữ nhật nhưng màu của khung sẽ cùng màu với cửa sổ hiện tại. Màu mặc định sẽ là màu trắng.
$SS_WHITERECT	Label sẽ có dạng một hình chữ nhật với màu nền sẽ trùng với màu nền của cửa sổ hiện tại. Mặc định màu nền sẽ là màu trắng.

Ví dụ : *#include <StaticConstants.au3>*

GUICreate("Student")

GUICtrlCreateLabel("Khoa ki thua Cong nghe - Moi truong", 10, 30, 100, 100, $SS_LEFTNOWORDWRAP)

GUISetState()

Sleep(3000)

2. GUICtrlCreatePic(filename, left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một control dùng để hiển thị ảnh.

- Bảng thông số hàm :

filename	Đường dẫn đến tập tin cần nạp : hỗ trợ các định dạng như BMP, JPG, GIF (nhưng không hỗ trợ ảnh động).

left	Tạo độ (X) bên trái của control.
top	Tọa độ (Y) tọa độ đỉnh trên của control.
width	Độ dài chiều ngang của control. Mặc định sẽ là độ dài chiều ngang đã được dùng trước đó.
height	Độ dài chiều cao của control. Mặc định sẽ là độ dài chiều cao được dùng trước đó.
style	Định nghĩa kiểu style của control. Xem thêm bảng style của hàm GUICtrlCreateLabel Mặc định (-1) : $SS_NOTIFY Style bắt buộc : $SS_BITMAP
exstyle	Tham số style mở rộng của control. Xem bảng exstyle của hàm GUICreate.

- Hàm này nếu gọi thành công sẽ trả về số ID của control này.

Ví dụ :

```
#include <StaticConstants.au3>
GUICreate("My best friend", 340, 255)
GUICtrlCreatePic("E:\Baoyen.jpg", 5, 5, 326, 244)
GUISetState()
Sleep(3000)
```

3. GUICtrlCreateButton("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một nút nhấn (button).

- Bảng thống số hàm

text	Văn bản sẽ xuất hiện trên nút bấm.
left	Vị trí bên trái của control (tọa độ X).

top	Vị trí đỉnh trên của control (tọa độ Y).
width	Độ dài chiều ngang của control. Mặc định sẽ là vừa khít với nội dung văn bản.
height	Độ dài chiều cao của control. Mặc định sẽ là vừa khít với chiều cao của nội dung văn bản.
style	Style cho button. Xem bảng style_button. Mặc định (-1) : không áp dụng style. Style bắt buộc : $WS_TABSTOP
exstyle	Định nghĩa style mở rộng cho control. Xem thêm bảng exstyle của hàm GUICreate.

Bảng style_button (tham khảo AutoIt Help)

Style Button	**Mô tả**
Để sử dụng các tham số style sau đây bạn phải thêm chỉ thị #include <ButtonConstants.au3> ngay tại phần đầu của chương trình.	
$BS_BOTTOM	Đặt nội dung văn bản nằm ở vị trí bên dưới (bottom) của button.
$BS_CENTER	Nội dung văn bản sẽ canh giữa theo chiều ngang trong button.
$BS_DEFPUSHBUTTON	Tạo một nút nhấn button với đường đóng khung có màu đen. Nếu nút button nằm trong hộp thoại, người dùng có thể chọn button bằng cách nhấn phím ENTER, ngay cả khi không có con trỏ nhập liệu. Style này thường dùng cho button nhằm mục đích giúp cho người dùng nhanh chóng chọn lựa những tùy chọn thường dùng hoặc mặc định.
$BS_MULTILINE	Nếu nội dung văn bản trong button quá chiều dài của button thì nội dung văn bản không đủ chứa sẽ di chuyển xuống dòng mới.
$BS_TOP	Đặt nội dung văn bản ngay bên trên (top) của button.

$BS_VCENTER	Nội dung văn bản sẽ được canh giữa theo chiều dọc của button.
$BS_ICON	Chỉ định rằng các button sẽ hiển thị một icon.
$BS_BITMAP	Chỉ định rằng các button sẽ hiển thị một ảnh bitmap.
$BS_FLAT	Các button sẽ có dạng hai chiều. Nó không sử dụng các bóng đổ (shading) để tạo ảnh ba chiều.
$BS_NOTIFY	Cho phép một button gửi một thông điệp BN_KILLFOCUS và BN_SETFOCUS đến cửa sổ cha của nó. Lưu ý rằng các button gửi thông điệp BN_CLICKED cho dù nó có phong cách này. Để nhận được thông điệp BN_DBLCLK, các button phải có style BS_RADIOBUTTON hoặc BS_OWNERDRAW.

- Hàm này nếu gọi thành công sẽ trả về số ID của control này.

Ví dụ : *#include <ButtonConstants.au3>*

GUICreate("Test function", 340, 255)

GUICtrlCreateButton("Thanh pho tinh yeu va noi nho", 5, 5, -1, -1, $BS_CENTER)

GUISetState()

Sleep(3000)

4. GUICtrlCreateInput("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một khung textbox cho người dùng nhập văn bản vào.

- Bảng thông số hàm :

text	Văn bản hiển thị trong control.
left	Vị trí bên trái (tọa độ X) của control.
top	Vị trí đỉnh trên (tọa độ Y) của control.
width	Độ dài chiều ngang của control. Mặc định sẽ là dùng độ dài chiều ngang được dùng trước đó.
height	Độ dài chiều cao của control. Mặc định sẽ là dùng độ dài chiều cao được dùng trước đó.

style	Thiết lập style cho control. Xem thêm bảng Style. Mặc định (-1) : $ES_LEFT, $ES_AUTOHSCROLL. Style bắt buộc : $WS_TABSTOP chỉ khi không có style $ES_READONLY.
exstyle	Định nghĩa style mở rộng cho control. Bạn tham khảo thêm bảng exstyle của hàm GUICreate.

Bảng style cho control input (tham khảo AutoIt Help)

Style	Mô tả
Để sử dụng các tham số style sau đây bạn phải thêm chỉ thị #include <EditConstants.au3> ngay tại phần đầu của chương trình	
$ES_AUTOHSCROLL	Tự động di chuyển văn bản sang phải 10 kí tự khi nội dung văn bản mà người dùng gõ đang nằm ở vị trí cuối dòng. Khi người dùng nhấn phím Enter, control sẽ di chuyển nội dung văn bản trở về vị trí zero.
$ES_AUTOVSCROLL	Di chuyển nội dung văn bản lên một trang khi người dùng nhấn phím Enter tại dòng cuối cùng.
$ES_CENTER	Canh giữa nội dung văn bản trong một control có nhiều dòng.
$ES_LOWERCASE	Chuyển đổi tất cả các kí tự mà người dùng gõ vào thành chữ thường.
$ES_NUMBER	Chỉ cho phép người dùng nhập liệu kí tự dạng số.
$ES_OEMCONVERT	Style này thường dùng cho control chứa đường dẫn tập tin.
$ES_MULTILINE	Cho phép control được nhập liệu trên nhiều dòng. Mặc định chỉ là một dòng mà thôi.
$ES_PASSWORD	Hiển thị tất cả kí tự mà người dùng gõ vào thành dấu *. Thường dùng cho control nhập nội dung mật khẩu.
$ES_READONLY	Ngăn chặn người dùng gõ hoặc chỉnh sửa nội dung văn bản. (Chỉ đọc).

$ES_RIGHT	Nội dung văn bản sẽ được canh phải cho control hỗ trợ chế độ nhập liệu nhiều dòng (multiline).
$ES_UPPERCASE	Chuyển đổi tất cả kí tự mà người dùng gõ vào thành chữ hoa.
$ES_WANTRETURN	Thêm kí tự về đầu dòng khi người dùng nhấn phím ENTER và control hỗ trợ nhập liệu trên nhiều dòng. Nếu bạn không dùng style này thì khi nhấn phím ENTER sẽ có tác động tương tự như nhấn nút mặc định của hộp thoại. Style này sẽ không có tác dụng đối với các control chỉ hỗ trợ nhập liệu một dòng.
Tham khảo thêm style $ES_NOHIDESEL trong AutoIt Help.	

- Hàm này nếu gọi về thành công sẽ trả về số ID của control này.

Ví dụ : *#include <EditConstants.au3>*

GUICreate("My best friend", 340, 255)

GUICtrlCreateInput("levientrinh", 12, 12, 100, 50, $ES_PASSWORD)

GUISetState()

Sleep(3000)

5. GUICtrlCreateEdit("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một textbox để nhập nội dung văn bản, tuy nhiên sẽ hỗ trợ chế độ nhập liệu nhiều dòng.

- Bảng thông số hàm

text	Nội dung văn bản trong control.
left	Vị trí bên trái (tọa độ X) của control.
top	Vị trí đỉnh trên (tọa độ Y) của control.
width	Độ dài chiều ngang của control. Mặc định sẽ là dùng độ dài chiều ngang được dùng trước đó.
height	Độ dài chiều cao của control. Mặc định sẽ là dùng độ dài chiều cao được dùng trước đó.

style	Thiết lập style cho control. Mặc định (-1) : $ES_WANTRETURN, $WS_VSCROLL, $WS_HSCROLL, $ES_AUTOVSCROLL, $ES_AUTOHSCROLL Style bắt buộc : $ES_MULTILINE, $WS_TABSTOP chỉ khi không có style $ES_READONLY Xem bảng style trong hàm GUICtrlCreateInput.
exstyle	Định nghĩa style mở rộng cho control.

- Hàm này nếu gọi về thành công sẽ trả về số ID của control này.

Ví dụ : *#include <EditConstants.au3>*

GUICreate("My best friend", 340, 255)

GUICtrlCreateEdit("23071991", 12, 12, 100, 50, $ES_NUMBER)

GUISetState()

Sleep(3000)

6. GUICtrlCreateCheckbox("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo hộp kiểm checkbox cho người dùng lựa chọn.

- Bảng thông số hàm

text	Văn bản trong control checkbox.
left	Vị trí bên trái của control (tọa độ X).
top	Vị trí đỉnh trên của control (tọa độ Y).
width	Độ dài chiều ngang của control. Mặc định sẽ là vừa đủ.
height	Độ dài chiều cao của control. Mặc định sẽ là vừa đủ.
style	Xác lập style cho control. Mặc định (-1) : $BS_AUTOCHECKBOX. Xem bảng style phía dưới. Style bắt buộc : $WS_TABSTOP và $BS_AUTOCHECKBOX nếu checkbox không thiết lập style.
exStyle	Xác lập style mở rộng cho control. Xem bảng exstyle của hàm GUICreate.

Bảng style cho control checkbox (tham khảo AutoIt Help)

Style	**Mô tả**

Để sử dụng các tham số style sau đây, bạn phải thêm chỉ thị #include <ButtonConstants.au3> ngay tại phần đầu của chương trình.	
$BS_3STATE	Tạo một checkbox không thể đánh dấu chọn hoặc hủy chọn hộp kiểm.
$BS_AUTO3STATE	Tạo một hộp checkbox mà có trạng thái sẽ thay đổi lần lượt khi người dùng dùng chuột đánh dấu vào hộp chọn. Trạng thái sẽ thay đổi lần lượt từ chọn (✓), đánh dấu đen (■) hoặc không chọn.
$BS_AUTOCHECKBOX	Tạo một hộp checkbox sẽ có trạng thái lần lượt là chọn hoặc không chọn khi người dùng nhấn chuột vào checkbox.
$BS_CHECKBOX	Tạo ra một checkbox nhỏ và ở trạng thái không được chọn, nhãn văn bản sẽ được nằm ở bên phải. Để hiển thị văn bản nằm bên trái checkbox, bạn hãy kết hợp với style BS_RIGHTBUTTON.
$BS_LEFT	Văn bản sẽ hiển thị bên phải của hộp checkbox và sẽ được canh trái.
$BS_PUSHLIKE	Style này làm cho checkbox giống như một nút nhấn button. Khi bạn nhấn nó, nút sẽ chìm xuống, khi bạn nhấn nó lại một lần nữa nó sẽ nổi lên.
$BS_RIGHT	Văn bản sẽ hiển thị bên phải của hộp checkbox và sẽ được canh phải.
$BS_RIGHTBUTTON	Vị trí của checkbox sẽ nằm bên phải của văn bản trong tham số text.
$BS_GROUPBOX	Tạo ra một nhóm groupbox để chứa các button khác. Tham số text trong kiểu style này sẽ nằm ở góc trên bên trái của nhóm.
$BS_AUTORADIOBUTTON	Style này sẽ làm cho checkbox tương tự như các nút radiobutton. Trừ trường hợp người sử dụng chọn, nút chekcbox sẽ được nổi bật lên và tự động loại bỏ các lựa chọn từ các nút radio button khác trong cùng một nhóm có cùng kiểu style.

- Hàm này nếu gọi về thành công sẽ trả về số ID của control này.

Ví dụ : *#include <ButtonConstants.au3>*

GUICreate("Options", 340, 255)

GUICtrlCreateCheckbox("Edit", 12, 12, 200, 150, $BS_GROUPBOX)

GUISetState()

Sleep(8000)

7. GUICtrlCreateRadio("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một nút nhấn radio tương tự như chẹckbox, tuy nhiên nó có dạng hình tròn.

- Bảng thông số hàm

text	Văn bản hiển thị của control.
left	Vị trí bên trái của control (tọa độ X).
top	Vị trí đỉnh trên của control (tọa độ Y).
width	Độ dài chiều ngang của control. Mặc định sẽ là tự động vừa với nội dung
height	Độ dài chiều cao của control. Mặc định sẽ là tự động vừa với nội dung
style	Thiết lập style cho control Mặc định (-1) : không có áp dụng style nào cả. Style bắt buộc : $BS_AUTORADIOBUTTON và $WS_TABSTOP nếu nút nhấn Radio đầu tiên nằm trong một nhóm. Xem bảng style của hàm GUICtrlCreateButton.
exstyle	Thiết lập style mở rộng cho control. Xem bảng exstyle của hàm GUICreate.

- Hàm này nếu gọi thành công sẽ trả về số ID của control đó.

Ví dụ : *#include <ButtonConstants.au3>*

GUICreate("Options", 340, 255)

GUICtrlCreateRadio("Edit", 12, 12, 200, 150)

GUISetState()

Sleep(8000)

8. GUICtrlCreateDate("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một control date để bạn chọn ngày.

- Bảng thông số hàm

text	Ngày chọn mặc định đầu tiên của control (luôn có định dạng là "yyyy/mm/dd").
left	Vị trí bên trái của control (tạo độ x).
top	Vị trí đỉnh trên của control (tọa độ Y).
width	Độ dài chiều ngang của control. Mặc định là độ dài được dùng trước đó.
height	Độ dài chiều cao của control. Mặc định là độ dài được dùng trước đó.
style	Thiết lập style cho control. Xem bảng style phía dưới. Mặc định (-1) : $DTS_LONGDATEFORMAT Style bắt buộc : $WS_TABSTOP
exstyle	Thiết lập style mở rộng cho control. Xem bảng exstyle của hàm GUICreate. Mặc định (-1) : WS_EX_CLIENTEDGE

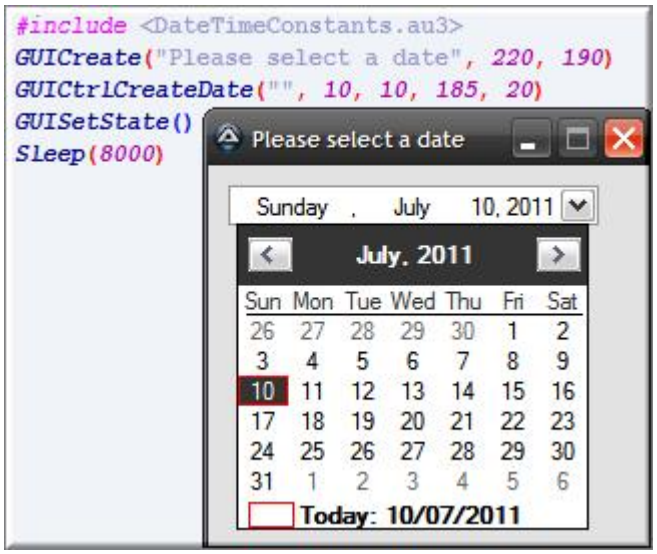
Bảng style cho control Date (tham khảo AutoIt Help)

Style	**Mô tả**
Để sử dụng tham số style sau đây, bạn phải thêm chỉ thị #include <DateTimeConstants.au3> ngay tại phần đầu của chương trình.	
$DTS_UPDOWN	Control sẽ có dạng một hộp updown box (có nút để tăng và giảm giá trị thời gian). Style này được dùng thay vì phải dùng một dạng style đổ xuống để chọn thời gian trực tiếp vốn là style mặc định.
$DTS_SHOWNONE	Cho phép control không có lựa chọn ngày (có một hộp checkbox bên cạnh ngày để chọn).
$DTS_LONGDATEFORMAT	Ngày tháng sẽ hiển thị ở định dạng đầy đủ. Định dạng chuỗi mặc định sẽ được quy định trong tham số LOCALE_SLONGDATEFORMAT Thường sẽ có hình thức như "Friday, April 19 1998".
$DTS_TIMEFORMAT	Style này dùng để hiển thị thời gian. Định dạng của thời gian được quy định trong tham số LOCALE_STIMEFORMAT, thường có hình thức giống như "5:31:42 PM."

$DTS_RIGHTALIGN	Hộp đổ xuống chọn ngày tháng sẽ nằm bên trái chỗ nút combobox.
$DTS_SHORTDATEFORMAT	Hiển thị thời gian dưới dạng chuỗi ngắn. Định dạng chuỗi còn phụ thuộc vào tham số LOCALE_SSHORTDATE, thường là có định dạng như "4/19/96".

- Hàm này nếu gọi thành công sẽ trả về số ID của control đó.

Ví dụ :



9. GUICtrlCreateGraphic(left, top, [width], [height], [style])

- Công dụng : tạo một khu vực graphic (đồ họa) dùng để hiển thị các đối tượng hình học trên một cửa sổ GUI nào đó. Bạn sẽ hiểu thêm control này trong các bài viết kì tới về sử dụng công cụ Koda FormDesigner để thiết kế giao diện.

- Bảng thông số hàm

left	Tọa độ trái của control (tọa độ X)
top	Tọa độ đỉnh trên của control (tọa độ Y)
width	Độ dài chiều ngang của control. Mặc định là độ dài chiều ngang được dùng trước đó.
height	Độ dài chiều cao của control. Mặc định là độ dài chiều cao được dùng trước đó.
style	Thiết lập style cho control. Xem thêm bảng style của hàm GUICreateLabel Mặc định (-1) : $SS_NOTIFY.

- Hàm này nếu gọi thành công sẽ trả về số ID tương ứng với control đó.

Ví dụ : *#include <StaticConstants.au3>*

GUICreate("Create Graphic", 220, 190)

GUICtrlCreateGraphic(20, 50, 100, 100, $SS_BLACKFRAME)

GUICtrlSetBkColor(-1, 0xffffff); thiết lập màu nền cho khu vực đồ họa là màu trắng. Bạn hãy tìm hiểu thêm hàm GUICtrlSetBkColor trong các bài viết sau.

GUISetState()

Sleep(8000)

10. GUICtrlCreateIcon(filename, iconname, left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một control dùng để chứa icon.

- Bảng thông số hàm

filename	Đường dẫn đến tập tin làm icon.
iconname	Tên icon cần lấy làm icon trong trường hợp filename có nhiều icon bên trong. Có thể dùng số để chỉ vị trí icon trong tập tin cần lấy icon. Mặc định là -1
left	Tọa độ bên trái của control (tọa độ X).
top	Tọa độ đỉnh trên của control (tọa đọ Y).
width	Độ dài chiều ngang của control. Mặc định là 32.
height	Độ dài chiều cao của control. Mặc định là 32.
style	Thiết lập style cho control. Xem tham số style trong hàm GUICtrlCreateLabel. Mặc định (-1) : $SS_NOTIFY Style bắt buộc : $WS_TABSTOP, $SS_ICON
exstyle	Định nghĩa style mở rộng cho control. Xem tham số exstyle trong hàm GUICreate.

Ví dụ : *#include <StaticConstants.au3>*

GUICreate("Icon", 220, 190)

GUICtrlCreateIcon("shell32.dll", 7, 20, 75, 32, 32)

GUISetState()

Sleep(8000)

11. GUICtrlCreateMonthCal("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một control dùng để hiển thị lịch theo tháng.

- Bảng thông số hàm

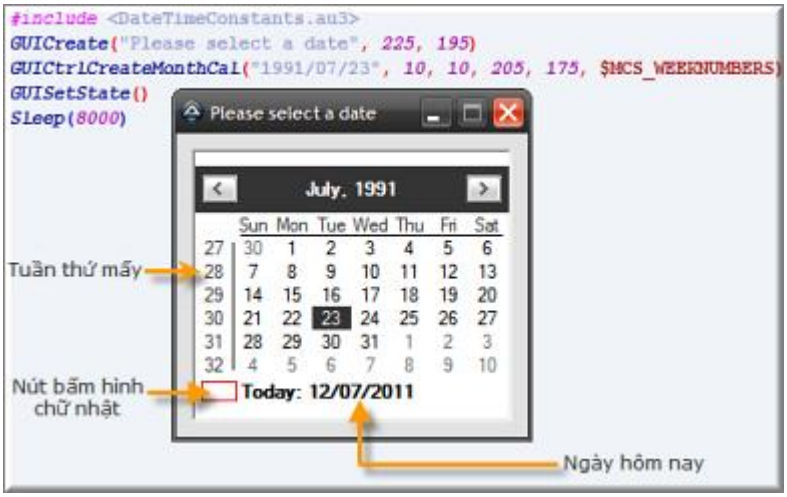
text	Ngày chọn mặc định khi control được hiển thị (luôn phải có định dạng "yyyy/mm/dd").
left	Tọa độ bên trái của control. (Tọa độ X).
top	Tọa độ đỉnh trên của control. (Tọa độ Y).
width	Độ dài chiều ngang của control. Mặc định là độ dài chiều ngang được dùng trước đó.
height	Độ dài chiều cao của control. Mặc định là độ dài chiều cao được dùng trước đó.
style	Thiết lập style cho control. Xem bảng style phía dưới. Mặc định (-1) : không thiết lập style Style bắt buộc : $WS_TABSTOP
exstyle	Định nghĩa style mở rộng cho control. Xem bảng exstyle của hàm GUICreate. Mặc định (-1) : WS_EX_CLIENTEDGE

Bảng tham số style

Style	Mô tả
Bạn hãy thên chỉ thị #include <DateTimeConstants.au3> ngay tại phần đầu chương trình để sử dụng tham số style sau	
$MCS_NOTODAY	Thông thường control hiển thị lịch tháng này sẽ hiển thị ngày tháng hôm nay ngay tại phía dưới lịch tháng. Tuy nhiên nếu bạn thiết lập style này thì control sẽ không hiển thị ngày tháng hôm nay ngay tại phía dưới control.
$MCS_NOTODAYCIRCLE	Style này sẽ không hiển thị nút bấm hình chữ nhật bên trái ngày hiện tại hôm nay (ở dưới control).

$MCS_WEEKNUMBERS	Style này sẽ hiển thị tuần thứ mấy trong một năm (có giá trị từ 1 đến 52).

Với dụ :



12. GUICtrlCreateList("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một danh sách để chứa thông tin. Tuy nhiên danh sách chỉ có một cột nhưng có nhiều dòng.

- Bảng thông số hàm

text	Văn bản sẽ hiển thị trong Control.
left	Vị trí bên trái của control (tọa độ X).
top	Vị trí đỉnh trên của control (tọa độ Y).
width	Độ dài chiều ngang của control. Mặc định là độ dài chiều ngang được dùng trước đó.
height	Độ dài chiều cao của control. Mặc định là độ dài chiều cao được dùng trước đó.
style	Thiết lập style cho control. Xem bảng style phía dưới. Mặc định (-1) : $LBS_SORT, $WS_BORDER, $WS_VSCROLL Style bắt buộc : $WS_TABSTOP, $LBS_NOTIFY
exstyle	Style mở rộng cho control. Xem bảng tham số exstyle trong hàm GUICreate.

- Hàm này nếu gọi thành công sẽ trả về số ID tương ứng của control này.

- Để đọc giá trị được chọn trong List, dùng hàm GUICtrlRead.

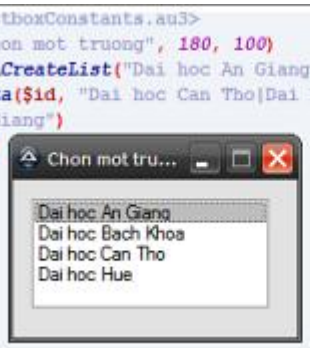
- Để thiết lập dữ liệu từng dòng trong List, xem hàm GUICtrlSetData.

Bảng style cho control List

Style	Mô tả
Để sử dụng các tham số style sau đây, bạn phải thêm chỉ thị #include <ListboxConstants.au3> ngay tại phần đầu chương trình.	
$LBS_DISABLENOSCROLL	Vô hiệu hóa thanh trượt dọc khi control List không chứa đủ nội dung văn bản. Thông thường nếu control không đủ chứa nội dung, thì sẽ hiển thị thanh trượt dọc để hiển thị phần thông tin bị khuất đi.
$LBS_NOSEL	Người dùng có thể xem nội dung trong List nhưng không thể chọn chúng.
$LBS_SORT	Sắp xếp các nội dung văn bản trong List theo kí tự đầu tiên của từng dòng.
$LBS_USETABSTOPS	Control List có thể nhận ra các kí tự tab trong chuỗi dòng văn bản mà người dùng định nghĩa khi sử dụng hàm GUICtrlSetData hoặc GUICtrlCreateList (thông thường kí tự tab sẽ không nhận ra). Vị trí tab mặc định sẽ nằm ở đơn vị thứ 32 của hộp thoại. Một đơn vị hộp thoại sẽ bằng một phần tư của một đơn vị độ dài chiều ngang của hộp thoại.
Xem thêm các style khác như $LBS_STANDARD, $LBS_NOTIFY, $LBS_NOINTEGRALHEIGHT trong AutoIt Help.	

Ví dụ :

```
#include <ListboxConstants.au3>
GUICreate("Chon mot truong", 180, 100)
$id = GUICtrlCreateList("Dai hoc An Giang", 12, 12, 150, 80, $LBS_SORT)
GUICtrlSetData($id, "Dai hoc Can Tho|Dai hoc Bach Khoa|Dai hoc Hue", _
"Dai hoc An Giang")
GUISetState()
Sleep(8000)
```



13. GUICtrlCreateListView("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo control ListView dùng để chứa danh sách. Nó giống như control List tuy nhiên nó có nhiều cột.

- Bảng thông số hàm :

text	Tham số dùng để xác định các tiêu đề cột trong ListView. Bạn cung cấp tham số này theo dạng "headcol1|headcol2|headcol3|…|headcoln". (tiêu đề cột 1|tiêu đề cột 2|tiêu đề cột 3|…|tiêu đề cột thứ n).
left	Vị trí bên trái của control (tọa độ X).
top	Vị trí đỉnh trên của control (tọa độ Y).
width	Độ dài chiều ngang của control. Mặc định là độ dài chiều ngang được dùng trước đó.
height	Độ dài chiều cao của control. Mặc định là độ dài chiều cao được dùng trước đó.
style	Thiết lập style cho control. Mặc định (-1) : $LVS_SHOWSELALWAYS, $LVS_SINGLESE Style bắt buộc : $LVS_REPORT
exstyle	Thiết lấp style mở rộng cho control. Xem bảng exstyle của hàm GUICreate và bảng exstyle của ListView dưới đây.

Bảng style cho ListView

Style	**Mô tả**
Để sử dụng các tham số style sau đây, bạn phải thêm chỉ thị #include <ListviewConstants.au3> ngay tại phần đầu chương trình.	
$LVS_ICON	Tạo ListView với icon.
$LVS_REPORT	Chế độ xem ListView như là một báo cáo
$LVS_SMALLICON	Tạo ListView với một icon nhỏ
$LVS_LIST	Chỉ hiển thị ListView theo kiểu danh sách. Trong kiểu hiển thị này, chỉ có các dòng dữ liệu tại cột đầu tiên sẽ được hiển thị.
$LVS_EDITLABELS	Các dòng dữ liệu có thể được chỉnh sửa.

$LVS_NOCOLUMNHEADER	Tiêu đề cột sẽ không hiển thị trong chế độ xem báo cáo (report). Mặc định sẽ là hiển thị tiêu đề cột.
$LVS_NOSORTHEADER	Tiêu đề cột sẽ không có tác dụng khi nhấn vào nó. Style này thường áp dụng khi nhấp chuột vào một tiêu đề cột nhưng không thực hiện bất kì hành động nào, chẳng hạn như sắp xếp.
$LVS_SINGLESEL	Chỉ có một mục tại một thời điểm nào đó có thể được chọn.
$LVS_SHOWSELALWAYS	Nếu bạn chọn một phần nào đó trên control, nó sẽ được hiển thị.
$LVS_SORTASCENDING	Các mục nằm trong một cột sẽ được sắp xếp tăng dần.
$LVS_SORTDESCENDING	Các mục nằm trong một cột sẽ được sắp xếp giảm dần.
$LVS_NOLABELWRAP	Mục văn bản nằm trong một cột sẽ hiển thị chỉ ở trên một dòng trong chế độ xem icon.

Bảng exstyle ListView

Extended Styles	**Mô tả**
Để sử dung các tham số style sau đây, bạn phải thêm chỉ thị #include <ListviewConstants.au3> ngay tại phần đầu chương trình.	
$LVS_EX_FULLROWSELECT	Khi một mục văn bản nào đó được chọn, mục đó và tất cả các mục văn bản con sẽ được tô sáng.
$LVS_EX_GRIDLINES	Hiển thị một khung đường lưới xung quanh các mục (item). Lúc này bạn sẽ thấy nó giống như chế độ xem dữ liệu trong một bảng.
$LVS_EX_HEADERDRAGDROP	Style này cho phép người dùng sử dụng con chuột để thực hiện chế độ rê và thả (drag and drop) để tiến hành di chuyển một cột sang vị trí mới. Ví dụ : bạn muốn di chuyển cột 1 sang cột 2, thì bạn chỉ cần nhấn chọn cột 1 và di chuyển sang chỗ cột 2.

$LVS_EX_TRACKSELECT	Cho phép chế độ lựa chọn nóng theo kiểu hot-track. Trong chế độ này, khi bạn di chuyển lên một mục nào đó trong cột đầu tiên thì sau một khoản thời gian ngắn, thì mục đó sẽ được chọn tự động cho dù bạn có nhấp chuột hay không.
$LVS_EX_CHECKBOXES	Cho phép hiển thị checkbox (hộp chọn) tại mỗi mục trong cột đầu tiên.
$LVS_EX_BORDERSELECT	Nếu style này được thiết lập, thì khi một mục nào đó được chọn chỉ có màu đóng khung xung quanh mục đó được thay đổi thay vì được tô sáng như thường lệ.
$LVS_EX_FLATSB	Cho phép hiển thị thanh trượt trong trường hợp độ rộng hay chiều cao của control không chứa đủ dữ liệu để xem.
$LVS_EX_SUBITEMIMAGES	Cho phép một hình ảnh có thể hiển thị trong một mục con nào đó.

- Hàm này nếu gọi thành công sẽ trả về số ID tương ứng của control ListView tương ứng đó.

- Để tạo các mục dữ liệu nằm trong ListView, bạn hãy xem hàm GUICtrlListViewItem dưới đây

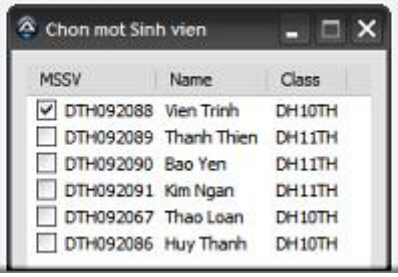
14. GUICtrlCreateListViewItem("text", listviewID)

- Công dụng : tạo các mục dữ liệu nằm trong control ListView.

- text là văn bản chứa các mục cần thêm. Mặc định sẽ dùng dấu | để ngăn cách các mục tương ứng với từng cột. listviewID là số ID của control ListView được trả về thông qua hàm GUICtrlCreateListView.

- Ví dụ chung cho hai hàm GUICtrlCreateListViewItem và hàm GUICtrlCreateListView

```
#include <ListviewConstants.au3>
GUICreate("Chon mot Sinh vien", 240, 140)
$listview = GUICtrlCreateListView("MSSV|Name|Class", 10, 10, 220, 130, _
$LVS_SINGLESEL, $LVS_EX_CHECKBOXES)
GUICtrlCreateListViewItem("DTH092088|Vien Trinh|DH10TH", $listview)
GUICtrlCreateListViewItem("DTH092089|Thanh Thien|DH11TH", $listview)
GUICtrlCreateListViewItem("DTH092090|Bao Yen|DH11TH", $listview)
GUICtrlCreateListViewItem("DTH092091|Kim Ngan|DH11TH", $listview)
GUICtrlCreateListViewItem("DTH092067|Thao Loan|DH10TH", $listview)
GUICtrlCreateListViewItem("DTH092086|Huy Thanh|DH10TH", $listview)
GUISetState()
Sleep(20000)
```



15. GUICtrlCreateCombo("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một hộp combobox chứa danh sách đổ xuống để lựa chọn một mục nào đó.

- Bảng thông số hàm

text	Văn bản sẽ xuất hiện trong control (chưa phải là các mục trong danh sách).
left	Vị trí bên trái của control (tọa độ X).
top	Vị trí đỉnh trên của control (tọa độ Y).
width	Độ dài chiều ngang của control. Mặc định là độ dài chiều ngang được dùng trước đó.
height	Độ dài chiều cao của control. Mặc định là độ dài chiều cao được dùng trước đó.
style	Thiết lập style mở rộng cho control. Xem bảng style phía dưới. Mặc định (-1) : $CBS_DROPDOWN, $CBS_AUTOHSCROLL, $WS_VSCROLL Bắt buộc : $WS_TABSTOP
exstyle	Thiết lập style mở rộng cho control. Xem bảng exstyle của hàm GUICreate.

Bảng style cho control Combobox (tham khảo AutoIt Help)

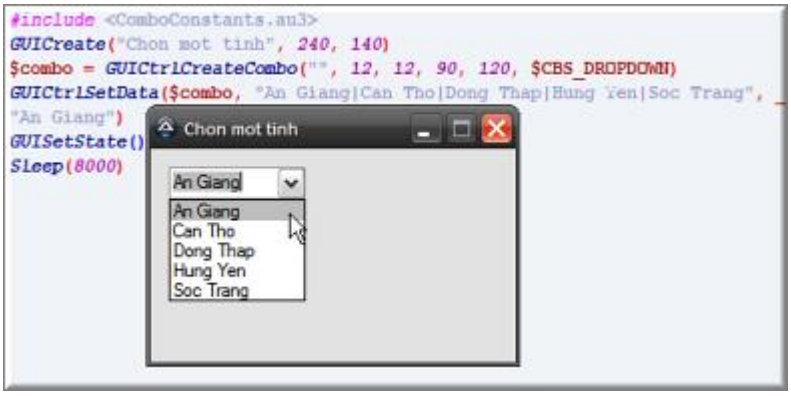
Style	**Mô tả**
Để sử dụng các tham số style sau đây, bạn phải thêm chỉ thị #include <ComboConstants.au3> ngay tại phần đầu chương trình.	
$CBS_AUTOHSCROLL	Tự động cuộn văn bản sang bên phải khi người dùng gõ nội dung văn bản đến cuối dòng. Nếu bạn không thiết lập style này, chỉ có nội dung văn bản vừa với khung Combobox sẽ được nhìn thấy.
$CBS_DISABLENOSCROLL	Vô hiệu hóa thanh trượt dọc (thanh trượt dọc xuất hiện khi số mục trong combobox quá dài so với chiều cao của nó). Nếu bạn không thiết lập style này, mặc định sẽ hiển thị đầy đủ nội dung các mục được thiết lập trong combo.
$CBS_DROPDOWN	Chỉ hiển thị control theo mặc định. Người dùng có thể chọn biểu tượng tam giác đổ xuống để thấy danh sách các mục chọn trong combobox.

$CBS_DROPDOWNLIST	Hiển thị nội dung trường văn bản tĩnh sẽ hiển thị mục được chọn trong danh sách.
$CBS_LOWERCASE	Các mục chọn trong combobox sẽ được chuyển đổi thành chữ thường hết.
$CBS_OEMCONVERT	Chuyển đổi kí tự do người dùng gõ vào từ kí tự Windows CE sang kí tự OEM sau đó trở lại với thiết lập kí tự Windows CE. Style này thường hữu ích cho các mục chọn có chứa tên tập tin. Nó chỉ áp dụng cho combobox đã thiết lập style $CBS_DROPDOWN.
$CBS_SIMPLE	Hiển thị toàn bộ danh sách các mục chọn kể cả khi người dùng chọn xong một mục chọn nào đó. Thông thường sau khi người dùng chọn xong một mục chọn thì hộp combobox sẽ thu lại chỉ còn lại một mục tương ứng được chọn.
$CBS_SORT	Sắp xếp các mục chọn trong combobox.
$CBS_UPPERCASE	Các mục chọn trong combobox sẽ được chuyển đổi thành chữ hoa hết.
Xem thêm style $CBS_NOINTEGRALHEIGHT trong AutoIt Help.	

- Để lấy giá trị của mục trong Control, xem hàm GUICtrlRead.
- Để thiết lập các mục chọn trong Combobox, xem hàm GUICtrlSetData.
- Hàm này nếu gọi thành công sẽ trả về số ID tương ứng với control đó.

Ví dụ :

```
#include <ComboConstants.au3>
GUICreate("Chon mot tinh", 240, 140)
$combo = GUICtrlCreateCombo("", 12, 12, 90, 120, $CBS_DROPDOWN)
GUICtrlSetData($combo, "An Giang|Can Tho|Dong Thap|Hung Yen|Soc Trang",
"An Giang")
GUISetState()
Sleep(8000)
```



16. GUICtrlCreateUpDown(inputcontrolID, [style])

- Công dụng : tạo control up-down (nút tăng giảm giá trị) bên cạnh các control nhập dữ liệu như Input, Edit,…
- inputcontrolID là số ID của các control nhập liệu. style là tham số quy định thiết lập

kiểu cho control. Mặc định (-1) : $UDS_ALIGNRIGHT. Style bắt buộc : $UDS_SETBUDDYINT và $UDS_ALIGNRIGHT nếu không định nghĩa canh biên. Xem bảng style phía dưới đây (tham khảo AutoIt Help)

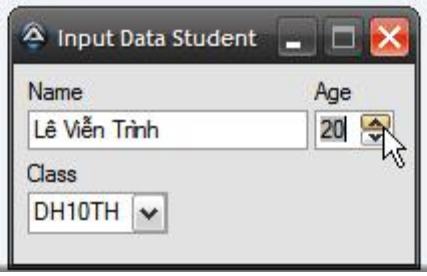
Style	Mô tả
Để sử dụng các tham số style sau đây, bạn phải thêm chỉ thị #include <UpdownConstants.au3> ngay tại phần đầu chương trình.	
$UDS_ALIGNLEFT	Vị trí của control UpDown sẽ nằm bên trái của các control Input
$UDS_ALIGNRIGHT	Vị trí của control UpDown sẽ nằm bên phải của các control Input
$UDS_ARROWKEYS	Bạn có thể sử dụng các phím muỗi tên ↑ hoặc ↓ để tăng giảm giá trị thay vì phải nhấn vào nút tăng giảm trong control UpDown.
$UDS_HORZ	Các nút tăng giảm trên control UpDown sẽ nằm ngang thay vì nằm dọc như thường lệ.
$UDS_NOTHOUSANDS	Không cho phép sử dụng dấu phân cách hàng nghìn trong con số.
Xem thêm style $UDS_WRAP trong AutoIt Help.	

- Giá trị Max và Min của giá trị có thể tăng, giảm tối đa có thể thiết lập thông qua hàm GUICtrlSetLimit.

- Hàm này nếu gọi thành công sẽ trả về số ID tương ứng với control đó.

Ví dụ :

```
#include <UpdownConstants.au3>
GUICreate("Input Data Student", 200, 100)
GUICtrlCreateLabel("Name", 7, 7)
GUICtrlCreateInput("", 7, 23, 140, 20)
GUICtrlCreateLabel("Age", 150, 7)
$s = GUICtrlCreateInput("", 150, 23, 40, 20)
GUICtrlCreateUpdown($s, $UDS_WRAP)
GUICtrlCreateLabel("Class", 7, 48)
$cb = GUICtrlCreateCombo("DH8TH", 7, 64, 70)
GUICtrlSetData($cb, "DH9TH|DH10TH|DH11TH", "DH10TH")
GUISetState()
Sleep(40000)
```



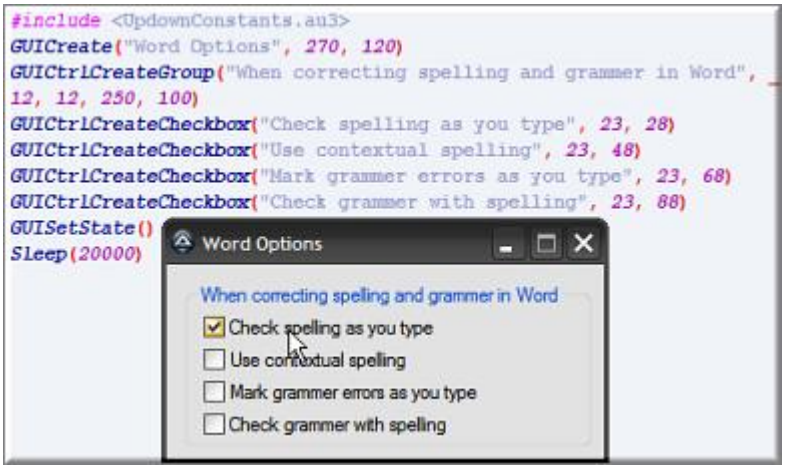
17. GUICtrlCreateGroup("text", left, top, [width], [height], [style], [exstyle])

- Công dụng : tạo một control Group, thường dùng để nhóm các đối tượng lại với nhau (áp dụng cho Checkbox hay Radio button).

- Bảng thông số hàm

text	Văn bản sẽ hiển thị trên Group.
left	Vị trí bên trái của control. (tọa độ X)
top	Vị trí đỉnh trên của control. (tọa độ Y)
width	Độ dài chiều ngang của control. Mặc định là độ dài chiều ngang được dùng trước đó.
height	Độ dài chiều cao của control. Mặc định là độ dài chiều cao được dùng trước đó.
style	Thiết lập style cho control. Xem bảng style của hàm GUICreate. Mặc định (-1) : không có style. Style bắt buộc : $WS_GROUP, $BS_GROUPBOX.
exstyle	Tham số quy định style mở rộng cho control. Xem bảng exstyle của hàm GUICreate.

Ví dụ :



Bạn có thể xem thêm các hàm tạo control khác như GUICtrlCreateAvi (control dùng để hiện thị tập tin video AVI), GUICtrlCreateSlider (tạo một thanh trượt tương tự như nút tăng giảm trong Volume Control, tuy nhiên nó nằm ngang), GUICtrlCreateContextMenu (tạo một thanh thực đơn khi nhấp phải vào một control hay một cửa sổ), GUICtrlCreateMenu, GUICtrlCreateMenuItem (tạo một menu và các mục con trong một mục menu), GUICtrlCreateTab, GUICtrlCreateTabItem, GUICtrlCreateTreeView, GUICtrlCreateTreeViewItem,…

Hàm cập nhật control

1. GUICtrlDelete(controlID)

- Công dụng : xóa một control được tạo trước đó.

- controlID là số ID của control cần xóa được trả về thông qua các hàm tạo control.

Ví dụ : *#include <UpdownConstants.au3>*

GUICreate("Word Options", 300, 300)

$gr = GUICtrlCreateGroup("Delete Group", _

12, 12, 250, 100)

GUICtrlDelete($gr)

GUISetState()

Sleep(20000)

2. GUICtrlSetBkColor(controlID, bkcolor)

- Công dụng : thay đổi màu nền hiện tại của một control sang một màu khác.

- controlID là số ID của control được trả về thông qua các hàm tạo control. bkcolor là mã màu mới mà bạn cần gán cho control.

- Chỉ có các control như Button, Label, Checkbox, Group, Radio, Edit, Input, List, ListView, ListViewItem, TreeView, TreeViewItem, Graphic, Progress, Slider và Combo là có thể thay đổi màu nền.

Ví dụ : *#include <ComboConstants.au3>*

GUICreate("Word Options", 300, 300)

$combo = GUICtrlCreateCombo("", 12, 12, 90, 120, $CBS_DROPDOWN)

GUICtrlSetData($combo, "An Giang|Can Tho|Dong Thap|Hung Yen|Soc Trang", _

"An Giang")

GUICtrlSetBkColor($combo, 0x00ff00); thiết lập màu nền xanh cho Combobox.

```
GUISetState()
Sleep(8000)
```

3. GUICtrlSetColor(controlID, color)

- Công dụng : thay đổi màu chữ cho control.

- controlID là số ID của control được trả về thông qua các hàm tạo control. color là mã màu mới mà bạn cần gán cho control.

- Chỉ có các control như Button, Label, Checkbox, Group, Radio, Edit, Input, List, ListView, ListViewItem, TreeView, TreeViewItem, Graphic, Progress và Combo là có thể thay đổi màu nền.

Ví dụ :
```
#include <ComboConstants.au3>
GUICreate("Word Options", 300, 300)
$lb = GUICtrlCreateLabel("An Giang University", 12, 12, 120, 90)
GUICtrlSetColor($lb, 0xff0000); thiết lập màu chữ đỏ cho Label.
GUISetState()
Sleep(8000)
```

4. GUICtrlSetData(controlID, data, [default])

- Công dụng : thiết lập dữ liệu cho control.

- controlID là số ID của control cần thiết lập dữ liệu được trả về từ hàm GUICtrlRead. data là dữ liệu mà bạn cầm muốn thiết lập. Bạn xem bảng sau để biết tham số data sẽ tác động đối với các control như thế nào

Control	Tác động
Combo, List, ListView, ListViewItem	Thay đổi các mục (item) trong control. Các mục thường sẽ được ngăn cách bởi dấu |
Progress	Thay đổi giá trị phần trăm của control Progress.
Slider	Thay đổi giá trị của thanh trượt Slider.
Group, Label, Button, Checkbox, Radio, Combo, List, Input, Edit, TabItem	Văn bản sẽ hiển thị trên các control này.
Date	Thay đổi ngày hoặc thời gian của control, tùy thuộc vào style của control này.

- Tham số default là tham số thiết lập dữ liệu mặc định cho control. Đối với control Combo, List sẽ là giá trị mặc định. Với các control Edit, Input : nếu bạn quy định tham số này không phải là chuỗi rỗng thì dữ liệu của tham số này sẽ được chèn vào vị trí con trỏ hiện hành trong control.

- Đối với control Combo hoặc List : nếu tham số data trùng với một mục đã có trong các control đó thì mục đó sẽ được gán mặc định. Nếu tham số data được bắt đầu bằng '|' (kí tự |) hoặc là chuỗi rỗng thì danh sách các mục trong các control đó sẽ bị hủy.

- Đối với các control như ListView và ListViewItem, để cập nhật tiêu đề cho một cột bạn cung cấp tham số như "||new_column", sẽ gán tên cột thứ ba là new_column. Nếu tham số new_column rỗng thì toàn bộ các cột và mục con trong cột của nó sẽ bị xóa. Ví dụ data = "|" sẽ xóa cột thứ hai, data = "" sẽ xóa cột thứ nhất.

Ví dụ : *#include <ListviewConstants.au3>*

GUICreate("Chon mot Sinh vien", 240, 140)

$listview = GUICtrlCreateListView("MSSV|Name|Class", 10, 10, 220, 130, _

$LVS_SINGLESEL, $LVS_EX_CHECKBOXES)

GUICtrlCreateListViewItem("DTH092088|Vien Trinh|DH10TH", $listview)

GUICtrlCreateListViewItem("DTH092089|Thanh Thien|DH11TH", $listview)

GUICtrlCreateListViewItem("DTH092090|Bao Yen|DH11TH", $listview)

GUICtrlSetData($listview, "||Lop"); thay đổi tiêu đề cột thứ ba Class thành Lớp

GUISetState()

Sleep(8000)

5. GUICtrlSetDefBkColor(defbkcolor, [winhandle])

- Công dụng : thiết lập màu nền mặc định cho các control trong cửa sổ GUI.

- defbkcolor : là mã màu nền cần gán làm mặc định cho tất cả control. winhandle là biến handle của cửa sổ được trả về của hàm GUICreate. Nếu tham số này không cung cấp, sẽ sử dụng cửa sổ được dùng trước đó.

Ví dụ : *#include <ListviewConstants.au3>*

GUICreate("Default Color", 240, 140)

GUICtrlSetDefBkColor(0xFF0000); thiết lập màu đỏ làm màu nền mặc định cho control.

GUICtrlCreateLabel("Mr. Trinh", 10, 5)

GUICtrlCreateRadio("DH10TH", 10, 35)

GUICtrlCreateButton("Yes", 10, 65)

GUISetState()

Sleep(8000)

6. GUICtrlSetDefColor(defcolor, [winhandle])

- Công dụng và tham số hàm tương tự như hàm GUICtrlSetDefBkColor tuy nhiên là áp dụng cho màu của văn bản trên các control.

Ví dụ : *#include <ListviewConstants.au3>*

GUICreate("Default Color", 240, 140)

GUICtrlSetDefColor(0xFF0000); thiết lập màu đỏ làm màu văn bản mặc định cho control.

GUICtrlCreateLabel("Mr. Trinh", 10, 5)

GUICtrlCreateButton("Yes", 10, 30)

GUISetState()

Sleep(8000)

7. GUICtrlSetFont(controlID, size, [weight], [attrib], [fontname], [quality])

- Công dụng : thiết lập font chữ văn bản cho một control.

- Bảng thông số hàm

controlID	Số ID của control được trả về thông qua hàm tạo control mà bạn muốn thiết lập font chữ cho control đó.
size	Kích thước của font chữ. Mặc định là 8.5
weight	Tham số quy định đậm độ đậm của font chữ (mặc định là 400 = bình thường).
attribute	Tham số quy định kiểu dáng văn bản. Tham số này là 2 : in nghiêng, 4 : gạch dưới, 8: gạch ngang văn bản. Bạn có thể áp dụng nhiều kiểu dáng hơn bằng cách dùng số tổng của chúng, chẳng hạn bạn quy định tham số này là 6 thì văn bản sẽ vừa in nghiêng và gạch dưới.
fontname	Tên font mà bạn muốn dùng.
quality	Chất lượng của font chữ. (mặc định là PROOF_QUALITY=2).

- Mặc định, các control sẽ dùng các font chữ được thiết lập trong hàm GUISetFont.

- Bạn có thể thiết lập tham số size có dấu chấm thập phân cũng được.

Ví dụ : *GUICreate("GUICtrlSetFont", 240, 140)*

$lb = GUICtrlCreateLabel("Mr. Trinh", 10, 5, 120)

GUICtrlSetFont($lb, 10.5, 600, 14, "Verdana")

GUISetState()

Sleep(8000)

8. GUICtrlSetTip(controlID, tiptext, ["title"], [icon], [options])

- Công dụng : thiết lập mẫu chú thích nhỏ (tip) sẽ được hiển thị khi bạn rê chuột vào control.

- Bảng thông số hàm

controlID	Số ID của control cần gán chú thích (tip) cho control.
tiptext	Nội dung của văn bản sẽ hiển thị khi con chuột rê ngang qua control.
title	Tiêu đề của mẫu chú thích. Bắt buộc IE5+.
icon	Tham số quy định kiểu icon nào sẽ xuất hiện kế bên tiêu đề của chú thích. Bắt buộc IE5+. 0 =không có icon, 1 = icon thông tin, 2 = icon cảnh báo, 3 = icon lỗi.

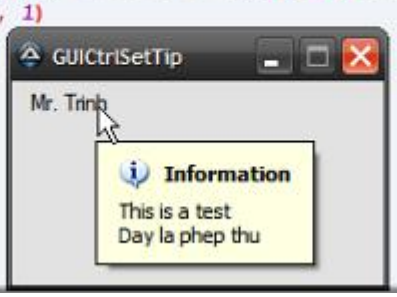
options	Tham số quy định mẫu chú thích sẽ hiển thị như thế nào. 1 = Hiển thị mẫu chú thích theo kiểu Balloon Tip. Bắt buộc IE5+ 2 = mẫu chú thích sẽ canh giữa theo chiều ngang của control.

- Chú thích sẽ hiển thị trong một khung hình chữ nhật có màu vảng.

- Bạn có thể dùng các kí tự như @CR hoặc @LF để tạo chú thích dạng nhiều dòng.

Ví dụ :

```
GUICreate("GUICtrlSetTip", 200, 110)
$lb = GUICtrlCreateLabel("Mr. Trinh", 10, 5, 120)
GUICtrlSetTip($lb, "This is a test"&@CR&"Day la phep thu", _
"Information", 1)
GUISetState()
Sleep(8000)
```



9. GUICtrlSetStyle(controlID, [style], [exstyle])

- Công dụng : thiết lập phong cách (style) cho một control.

- controlID là số ID của control được trả về từ hàm tạo control. style là tham số mà bạn muốn thiết lập cho control (xem bảng style của các hàm tạo control). exstyle là tham số thiết lập style mở rộng cho control (xem bảng exstyle của hàm GUICtrlCreate).

Ví dụ : *#include <StaticConstants.au3>*

GUICreate("GUICtrlSetStyle", 200, 110)

$lb = GUICtrlCreateLabel("Mr. Trinh", 10, 5, 120);

GUICtrlSetStyle($lb, $SS_RIGHT); thiết lập canh biên phải cho nội dung trong label

GUISetState()

Sleep(8000)

10. GUICtrlSetState(controlID, state)

- Công dụng : thiết lập trạng thái cho một control.

- control ID là số ID được trả về từ các hàm tạo control. state là tham số quy định trạng thái cho control. Xem bảng dưới đây để biết thêm tham số này.

State	Mô tả
No Change	0
$GUI_UNCHECKED	Radio, Checkbox hoặc ListViewItem sẽ không được đánh dấu check.
$GUI_CHECKED	Radio, Checkbox hoặc ListViewItem sẽ được đánh dấu check.

$GUI_INDETERMINATE	Checkbox sẽ có ba trạng thái.
$GUI_AVISTART	Control AVI sẽ trình diễn nội dung file AVI.
$GUI_AVISTOP	Ngừng chơi tập tin AVI trong control AVI
$GUI_AVICLOSE	Ngừng chơi tập tin AVI trong control AVI và giải phóng tài nguyên.
$GUI_DROPACCEPTED	Control sẽ chấp nhận thao tác rê và thả từ tập tin hoặc control khác.
$GUI_NODROPACCEPTED	Control sẽ không cho phép thao tác rê và thả (drag and drop).
$GUI_SHOW	Control sẽ hiển thị. Đối với control TabItem sẽ chọn thẻ đầu tiên để hiển thị.
$GUI_HIDE	Control sẽ bị ẩn (không xuất hiện).
$GUI_ENABLE	Control sẽ khả dụng cho các thao tác từ người dùng.
$GUI_DISABLE	Control sẽ bị vô hiệu hóa (lúc này control sẽ có màu xám).
$GUI_FOCUS	Control sẽ được nhận con nháy để nhập liệu hoặc được chọn.
$GUI_NOFOCUS	ListView sẽ bị mất con nháy được chọn.
$GUI_DEFBUTTON	Control sẽ được thiết lập như là một nút mặc định trên cửa sổ.
$GUI_EXPAND	Control TreeViewItem sẽ mở rộng các mục con nằm trong đó.
$GUI_ONTOP	Control sẽ có thuộc tính nổi (on top) đối với cửa sổ.

- Tham số state có thể kết hợp lại với nhau để có nhiều thuộc tính. Ví dụ như $GUI_DISABLE + $GUI_HIDE để có thể có thuộc tính vô hiệu hóa và bị ẩn đi.

Ví dụ :
```
#include <GUIConstantsEx.au3>
GUICreate("GUICtrlSetStyle", 200, 110)
$lb = GUICtrlCreateCheckbox("Nam", 10, 5)
$lb2 = GUICtrlCreateCheckbox("Nu", 10, 30)
GUICtrlSetState($lb, $GUI_DISABLE + $GUI_CHECKED)
GUISetState()
Sleep(8000)
```

11. GUICtrlSetPos(controlID, left, top, [width], [height])

- Công dụng : thay đổi vị trí và kích thước của một control.

- controlID là số ID của control cần thiết lập để thay đổi. left là tọa độ trái (tọa độ X) mới của control. top là tọa độ đỉnh trên (tọa độ Y) mới của control. width là độ dài chiều ngang mới của control. height là độ dài chiều cao mới của control.

Ví dụ :
```
#include <GUIConstantsEx.au3>
GUICreate("GUICtrlSetStyle", 200, 110)
$bt = GUICtrlCreateButton("Yes", 10, 10, 50, 20)
GUICtrlSetPos($bt, 100, 55, 60, 30)
GUISetState()
Sleep(8000)
```

12. GUICtrlSetOnEvent(controlID, "function")

- Công dụng : gọi một hàm do người dùng định nghĩa khi control được nhấn.

- controlID là số ID của control được trả về thông qua các hàm tạo Control. function là tên hàm sẽ cần gọi khi control được nhấn.

- Hàm "function" sẽ được gọi khi tham số tùy chọn GUIOnEventMode được gán bằng 1. Bạn có thể thêm dòng code Opt("GUIOnEventMode", 1) để thiết lập.

- Nếu tham số function là rỗng thì không có hàm nào được thực thi khi control được nhấn.

- Khi hàm được gọi thì chỉ số ID của hàm có thể được lấy về thông qua macro @GUI_CTRLID. Nếu muốn lấy thông tin handle của cửa sổ cũng như control bạn có thể lấy về tương ứng thông qua 2 macro @GUI_WINHANDLE và @GUI_CTRLHANDLE.

Ví dụ :
```
#include <GUIConstantsEx.au3>
Opt("GUIOnEventMode", 1)
GUICreate("GUICtrlSetOnEvent", 240, 80)
GUICtrlCreateLabel("Do you want to save the changes to "&"Baocao ?", 10, 10)
$btyes = GUICtrlCreateButton("Yes", 60, 40, 50, 20)
$btno = GUICtrlCreateButton("No", 120, 40, 50, 20)
GUICtrlSetOnEvent($btyes, "YesPressed")
GUICtrlSetOnEvent($btno, "NoPressed")
GUISetState()
Sleep(8000)
Func YesPressed()
   MsgBox(0, "Yes Pressed", "ID=" & @GUI_CtrlId & "CtrlHandle=" & @GUI_CtrlHandle)
EndFunc
Func NoPressed()
   MsgBox(0, "No Pressed", "ID=" & @GUI_CtrlId & "CtrlHandle=" & @GUI_CtrlHandle)
EndFunc
```

13. GUICtrlSetLimit(controlID, max, [min])

- Công dụng : dùng để giới hạn số kí tự hoặc con số đối với một số control.

- controlID là số ID của control được trả về thông qua hàm GUICtrlCreate… (hàm tạo các

control). max : đối với các control như Input hoặc Edit đó là số kí tự tối đa có thể nhập liệu trong control đó, đối với các control như Slider hoặc UpDown đó là con số tối đa có thể tăng tới được. min : mặc định là 0. Đối với các control như Input hoặc Edit đó là số kí tự tối thiểu phải có trong control đó, đối với các control như Slider hoặc UpDown đó là con số tối thiểu có thể giảm tới được.

Ví dụ : *#include <GUIConstantsEx.au3>*

Opt("GUIOnEventMode", 1)

GUICreate("GUICtrlSetLimit", 240, 80)

$ip = GUICtrlCreateInput("", 10, 10, 40, 20)

$ud = GUICtrlCreateUpdown($ip)

GUICtrlSetLimit($ud, 100, 10); thiết lập giá trị max/min cho con số tối đa có thể tăng giảm được cho control UpDown.

GUISetState()

Sleep(8000)

14. GUICtrlGetState([controlID])

- Công dụng : trả về thông tin trạng thái (state) của một control. Bạn xem các trạng thái state của hàm GUICtrlSetState để biết thêm các trạng thái.

- controlID là số ID của control mà bạn cần lấy trạng thái được trả về thông qua các hàm tạo control.

- Với control ListView, sẽ trả về số cột mà người dùng đã click.

Ví dụ : *#include <GUIConstantsEx.au3>*

GUICreate("GUICtrlGetState", 200, 140)

$lv = GUICtrlCreateListView("MSSV|Name|Class", 10, 10, 140, 80)

$bt = GUICtrlCreateButton("Yes", 45, 100, 60, 30)

GUISetState()

Sleep(6000)

MsgBox(0, "GUICtrlGetState", GUICtrlGetState($lv))

MsgBox(0, "GUICtrlGetState", GUICtrlGetState($bt))

15. GUICtrlRead(controlID, [advanced])

- Công dụng : đọc trạng thái hoặc dữ liệu của một control.

- controlID là số ID của control được trả về từ các hàm GUICtrlCreate… advanced : tham số tùy chọn quy định kết quả trả về chứa thông tin mở rộng của hàm. Nếu tham số này là 0, hàm sẽ trả về trạng thái hoặc dữ liệu của một control. Nếu tham số này là 1, hàm sẽ trả về thông tin mở rộng cho control. Mặc định là 0.

- Hàm này nếu gọi thành công sẽ trả về giá trị dữ liệu tùy thuộc vào từng loại control cụ thể. Xem bảng dưới đây.

Loại control	Giá trị trả về
Checkbox, Radio	Trạng thái của nút chọn. Xem bảng State trong hàm GUICtrlSetState.

Combo, List	Giá trị được chọn.
Input, Edit	Nội dung văn bản mà người dùng đã nhập.
Button	Văn bản hiển thị.
Date	Ngày được chọn trong control.
Progress	Tỉ lệ phần trăm hiện hành.
Slider	Giá trị hiện hành.
Tab	Một con số nào đó hoặc số ID của control tabitem được chọn còn phụ thuộc vào tham số advanced trong hàm.
Menu, MenuItem	Trạng thái của menu/item. Xem bảng State trong hàm GUICtrlSetState.
TreeView	Số ID của control TreeViewItem được chọn.
TreeViewItem	Trạng thái của TreeViewItem.
ListView	Số ID của control ListViewItem được chọn. Trả về 0 nếu không có mục nào được chọn.

Nếu tham số advanced = 1, thì hàm sẽ trả về những thông tin mở rộng được trình bày trong bảng sau.

Loại control	**Thông tin mở rộng của control sẽ trả về**
Checkbox, Radio	Văn bản của control.
Menu, MenuItem	Văn bản của control.
TreeView	Văn bản hiện hành được chọn trong TreeViewItem.
TreeViewItem	Văn bản của TreeViewItem.
ListViewItem	Trạng thái của ListViewItem nếu tham số style mở rộng (exstyle) $LVS_EX_CHECKBOXES được dùng. Xem bảng State trong hàm GUICtrlSetState.

Tab	Số ID của control tabitem.

Ví dụ : *#include <GUIConstantsEx.au3>*

Opt("GUIOnEventMode", 1)

GUICreate("GUICtrlGetData", 200, 140)

$ls = GUICtrlCreateList("Ke toan dai cuong", 10, 35, 120, 80)

$lb = GUICtrlCreateLabel("Chon mot mot hoc", 10, 10)

GUICtrlSetData($ls, "Kien truc may tinh|Xac suat thong ke|Toan cao cap|He dieu hanh")

GUICtrlSetOnEvent($ls, "Click")

GUISetState()

Sleep(8000)

Func Click()

MsgBox(0,"Mon hoc da chon", "Ban da chon mon "&GUICtrlRead($ls))

EndFunc

* ***Ghi chú*** : *việc tạo cửa sổ GUI hay control thực ra chúng ta không cần phải dùng lệnh code để tạo. Chúng ta có thể dùng công cụ Koda FormDesigner để thiết kế các control hoặc cửa sổ cho nhanh. Việc giới thiệu các hàm tạo control cũng như GUI giúp cho bạn thiết lập một số tham số đặc biệt và tối ưu cho các đối tượng đó. Sau đây, tôi xin giới thiệu bạn về công cụ thiết kế giao diện này.*

Hướng dẫn sử dụng Koda FormDesigner để thiết kế giao diện

- Lập trình giao diện với Koda FormDesigner

Trước đây, việc lập trình có giao diện đẹp là rất khó. Chúng ta đề ra một ví dụ nhỏ như lập trình có các nút textbox, combobox, label trong AutoIt để thiết kế các hộp nhập thông tin dành cho chương trình quản lý sinh viên là phải viết rất nhiều câu lệnh rồi, chưa nói đến việc lập trình có giao diện đẹp và đồng thời phải linh hoạt nữa.

Việc lập trình giao diện trong AutoIt với công cụ Koda FormDesigner là sử dụng những control có sẵn và bạn sẽ chỉ dùng chuột để kéo các control đó vào form để thiết kế, xác lập các thuộc tính cho chúng. Sau đó bạn chỉ sinh mã code tạo giao diện từ chương trình này, chép đoạn mã code vừa sinh ra qua cửa sổ SciTE để sử dụng. Tuy nhiên không giống như những công cụ lập trình giao diện khác, Koda FormDesigner chỉ thiết kế giao diện chứ không xử lý sự kiện. Nếu bạn muốn xử lý sự kiện thì sau khi chép đoạn mã code từ chương trình Koda FormDesigner qua SciTE, thì bạn phải cung cấp thêm các lệnh xử lý sự kiện trong SciTE.

- ***Giới thiệu Koda FormDesigner***

Koda FormDesigner là sản phẩm được phát triển bởi hai nhóm phát triển Martin Woods (Lookfar) và Dmitry Yudin (Lazycat) và một số thành viên khác. Hiện tại chương trình chỉ chạy trên môi trường Windows.

Trang chủ : http://koda.darkhost.ru

- ***Cấu hình máy để chạy Koda FormDesigner***

Chương trình này rất nhẹ nên bạn không cần lo lắng đến cấu hình máy để chạy chương trình này.

- ***Cài đặt Koda FormDesigner***

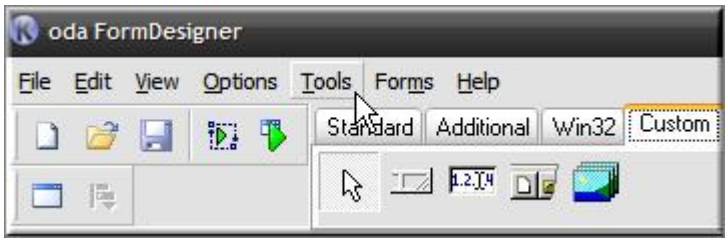
Bạn có thể vào trang http://koda.darkhost.ru/page.php?id=download để chọn cho mình

một phiên bản cài đặt phù hợp. Sau khi cài đặt, chương trình sẽ tích hợp vào menu của AutoIt trên trình đơn Start.

I. Tìm hiểu cửa sổ làm việc của chương trình và các hộp thoại

1. Main Window

Đây là cửa sổ chính của chương trình. Ở đây sẽ có thanh menu với các lệnh có sẵn trong Koda FormDesigner, thanh công cụ tùy chỉnh và các palete chứa các control có sẵn để bạn thiết kế giao diện. Palette được chia thành 4 tab, bao gồm : Standard, Additional, Win32, Custom. Để tạo một control mới trong form, bạn hãy dùng chuột click vào control mà bạn muốn dùng trong palette nào đó, sau đó kéo thả lên form, control sẽ được chèn.

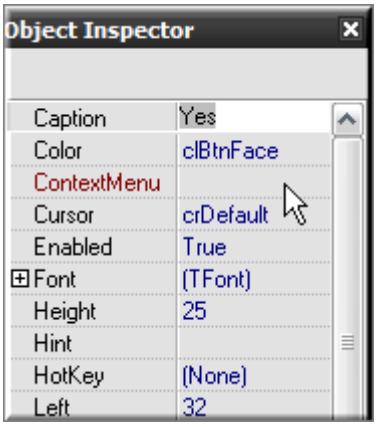


2. Object Inspector

Cửa sổ Object Inspector thường nằm ở dưới bên trái giao diện của chương trình. Cửa sổ này thường được chia thành hai phần. Phần đầu tiên là một Combobox chứa danh sách các form và control và phần thứ hai sẽ chứa khung chỉnh sửa thuộc tính cho form hoặc control.

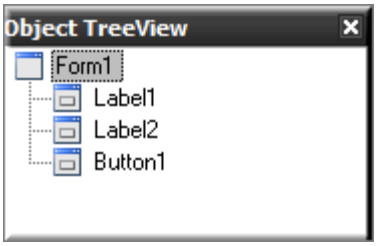
Khung combobox sẽ cho phép bạn truy cập mọi control trên form, cho dù nó bị ẩn (nằm dưới các control khác).

Trong thẻ Properties, bạn có thể thay đổi thuộc tính của control được chọn. Phần Properties được chia thành hai phần : tên thuộc tính và giá trị của chúng.



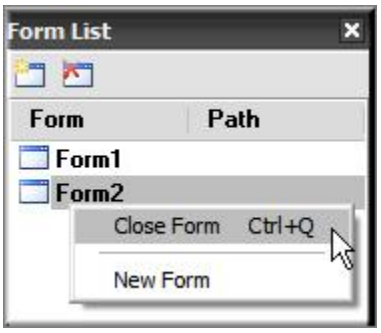
3. Object TreeView

Cửa sổ này sẽ hiển thị tất cả các đối tượng trong một form. Bạn có thể chuyển đổi giữa các đối tượng bằng cách chọn chúng trong cây. Ngoài ra bạn cũng dễ dàng chọn nhiều đối tượng bằng cách nhấn phím Ctrl. Bạn có thể chỉnh sữa tên đối tượng trong cửa sổ này. Tuy nhiên, tên bạn chỉnh sửa chỉ mang tính tượng trưng, bởi vì nó sẽ không phải là tên được sinh ra trong code tạo giao diện chính của chương trình (chỉ dùng thuộc tính Name trong cửa sổ **Object Inspector**).



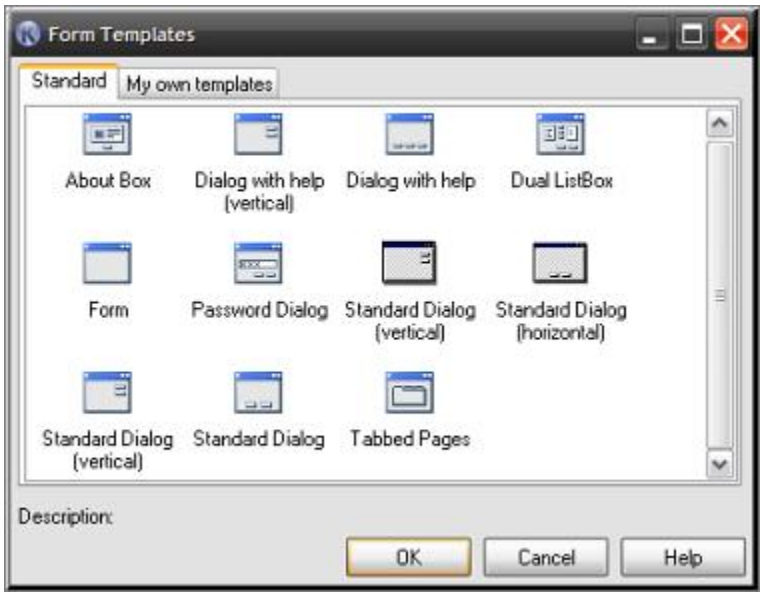
4. Form List

Cửa sổ này cho phép bạn xem danh sách tất cả các form được mở trong quá trình thiết kế. Bạn có thể chuyển đổi giữa các form được mở hoặc tạo một form mới, đóng một form đã có. Bạn có thể làm điều đó thông qua các nút bấm (button) hoặc qua menu chuột phải.



5. Templates Gallery

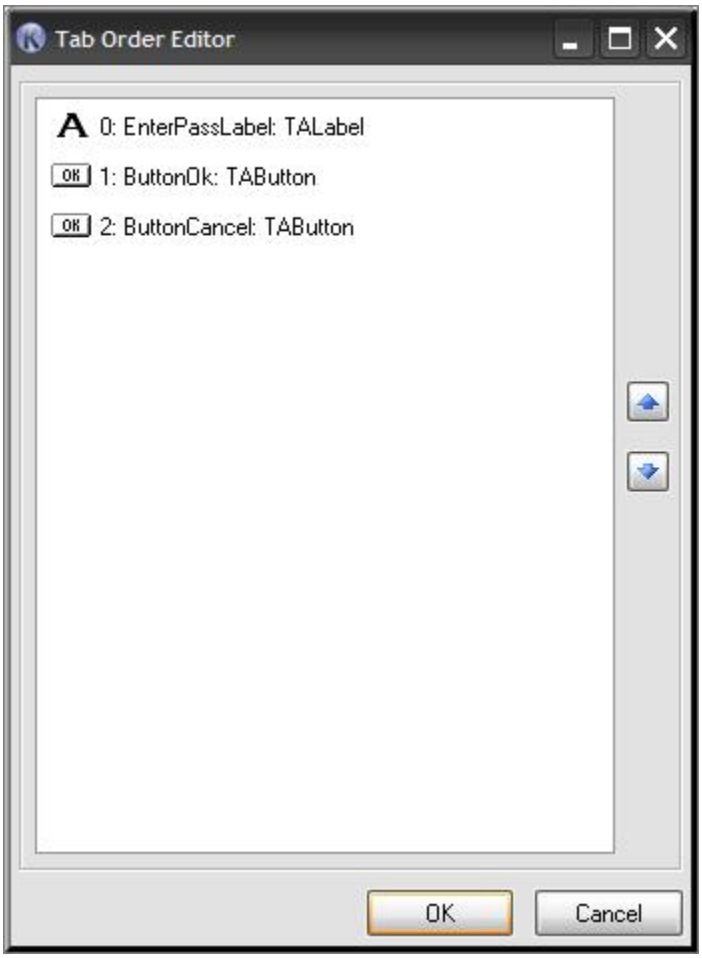
Đây là hộp thoại Templates dùng để cho phép bạn chọn các loại form dựa trên các Templates của chương trình. Hộp thoại này sẽ xuất hiện khi bạn nhấn chọn menu File -> New (Ctrl – N).



6. Tab Order Editor

Hộp thoại Tab Order Editor sẽ giúp bạn quy định vị trí thứ tự các control nhận con nháy mỗi khi bạn nhấn phím Tab (hộp thoại này xuất hiện khi bạn nhấn chuột phải vào một control và chọn Tab Order hoặc phím nóng Ctrl – T). Theo mặc định, AutoIt sẽ xác lập vị trí chuyển đổi con nháy đến các control theo thứ tự các control được tạo. Nếu bạn thay đổi vị trí Tab khi nhảy giữa các control sẽ ảnh hưởng đến đoạn code sinh ra bởi chương trình. Bạn có thể thay đổi vị trí trong hộp thoại này bằng cách dùng chuột nhấn vào một control sau đó thả nó đến một vị trí mong muốn hoặc bạn có thể dùng nút bấm mũi tên

lên xuống ngay tại vị trí bên phải của hộp thoại. Việc thay đổi vị trí control trong hộp thoại này sẽ ảnh hưởng đến thuộc tính TabOrder của control đó. Nếu bạn muốn, bạn có thể thay đổi thuộc tính này bằng tay.



7. Code Dialog

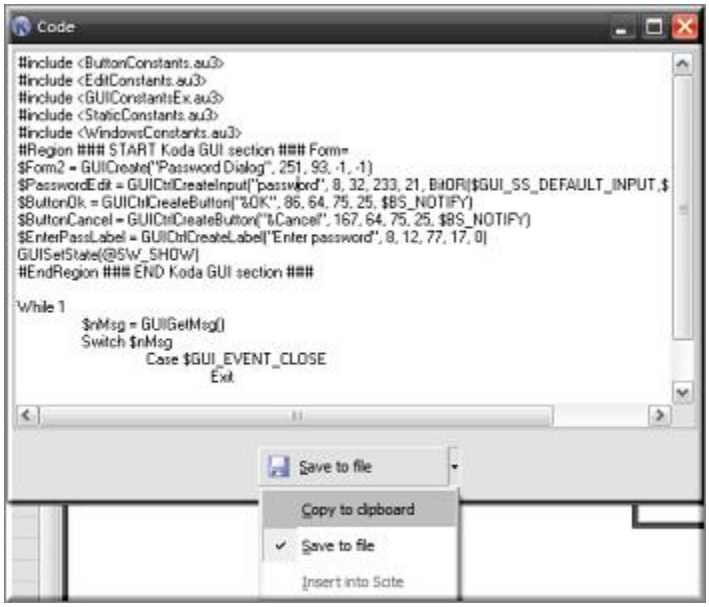
Hộp thoại này cung cấp cho bạn đoạn code sinh ra giao diện mà bạn đã thiết kế. Để thấy được hộp thoại này, ngay sau khi thiết kế giao diện xong bạn chọn Tools -> Generate Form Code (phím tắt F9) hoặc nhấn vào biểu tượng của nó trên thanh công cụ của chương trình.

Phím muỗi tên tam giác hướng xuống dưới cho phép bạn chọn ba chế độ save code mà chương trình cung cấp.

* Copy to clipboard (sao chép nội dung code sinh ra vào Clipboard)

* Save to file (lưu trữ thành tập tin có phần mở rộng là *.au3)

* Insert into SciTE. Chép đoạn code sinh ra vào trong cửa sổ SciTE Script Editor hiện hành. Tùy chọn này sẽ khả dụng nếu bạn gọi Koda FormDesigner từ SciTE Script Editor bằng cách chọn Tools -> Koda (FormDesigner), phím tắt Alt – m (tùy chọn menu này sẽ xuất hiện nếu bạn đang mở một tập tin *.au3).

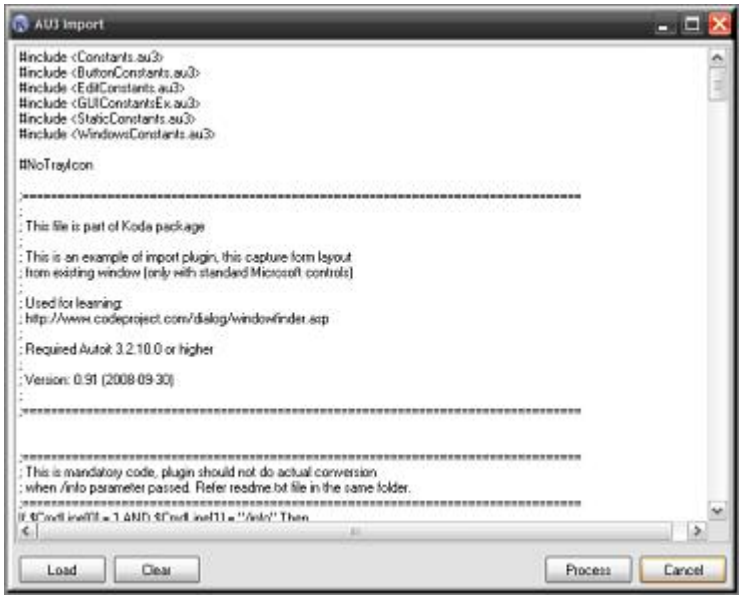


II. Tìm hiểu thanh menu của chương trình

1. File

Trong menu File có New : tạo một form mới dựa trên các Template sẵn có, Open : mở một tập tin form được thiết kế bởi Koda FormDesigner (*.kxf) hoặc tập tin script (*.au3), Close Form: đóng form thiết kế hiện hành, Save : lưu trữ form, Save as : lưu trữ form dưới một tên khác, Recent Files : xem danh sách những tập tin được mở gần đây (mặc định là 10, bạn vào Options -> Options, trong nhóm General, dưới mục Number of recent file list (1-20), bạn gõ số tập tin được chỉnh sửa gần đây cần muốn xem trong mục Recent).

Trong menu File có tùy chọn Import, trong mục này có mục Import AutoIt GUI. Lệnh này sẽ giúp bạn mở một hộp thoại để bạn có thể dán đoạn mã Script từ Clipboard hoặc nạp một tập tin *.au3 bằng cách dùng nút Load. Hộp thoại này sẽ giúp bạn chuyển đổi đoạn mã script tạo form nguồn thành một form trong Koda FormDesigner. Tất nhiên mỗi lần chuyển đổi sẽ chỉ là một tập tin nguồn trên một lần. Sự chuyển đổi này sẽ loại bỏ những phần không cần thiết cũng như trùng lặp trong script nguồn. Bạn có thể nhấn vào nút Clear để chuyển đổi sang một đoạn Script mới. Khi bạn hoàn thành xong, hãy nhấn Process đế cho chương trình tạo Form cho bạn.



2. Edit

Trong menut Edit có Undo : hủy bỏ thao tác trước đó (bạn có thể vào Options -> Options, trong nhóm General, ở mục Undo levels (0-disable), bạn có thể cấu hình số lần để Undo), Cut/Copy : cắt, sap chép đối tượng đến Clipboard, Paste : dán đối tượng từ Clipboard, Delete : xóa đối tượng được chọn, Select All : chọn tất cả control trong form hiện hành, Tab Order : mở hộp thoại Tab Order Editor, Align : các mục trong menu này giúp bạn thực hiện các thao tác canh biên đối với control hoặc một nhóm control.

Trong menu Edit có lệnh con Control, trong mục này bạn có thể gán một số hành động đối với control được chọn, chẳng hạn :

- Bring to front : đặt các control được chọn nằm ở trên control khác.

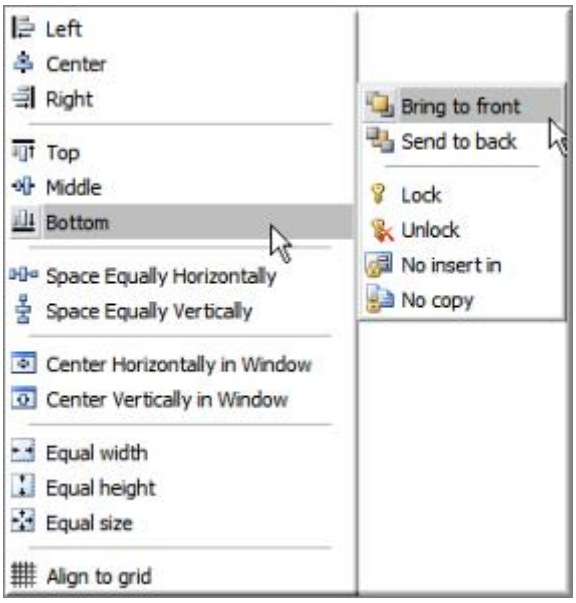
- Send to Back : đặt các control được chọn nằm bên dưới các control khác.

- Lock : khóa các control được chọn. Không cho phép di chuyển và/hoặc thay đổi kích thước.

- Unlock : mở khóa cho các control được chọn.

- No Copy : không cho sao chép đối tượng control được chọn.

- No insert in : không cho phép các đối tượng control được tạo ngay bên trong đối tượng control được áp dụng lệnh này. Thường áp dụng cho control dạng container (chứa).



3. View

Trong menu View có các mục như Object Tree, Object Inspector, Form List. Bạn nhấn chọn hoặc hủy chọn để hiện hoặc ẩn các cửa sổ tương ứng.

4. Options

Trong menu Options có hai mục Options và Remember position. Khi bạn chọn mục Options thì sẽ xuất hiện cửa sổ để bạn thiết lập các tùy chọn liên quan đến môi trường làm việc của Koda FormDesigner. Trong cửa sổ Options sẽ có các nhóm sau để bạn thiết lập.

** General*

- Form Run Method : phương thức để khởi chạy một form được tạo. Nó có thể là

■ Associated : lựa chọn này sẽ là lựa chọn mặc định. Bạn nên chọn tùy chọn này nếu bạn có cài đặt AutoIt trên máy tính của mình. Mỗi khi form mới được tạo trong Koda FormDesigner, khi bạn xem thử cửa sổ được tạo (nhấn F10) thì sẽ dùng AutoIt mặc định

trên máy để chạy nó.

■ Manual : bạn phải chọn đường dẫn đến tập tin thực thi của AutoIt để khởi chạy form được tạo trong Koda FormDesigner. Thông thường mặc định sẽ là C:\Program Files\AutoIt3\AutoIt3.exe.

- At Koda startup : nhóm tùy chọn này sẽ quy định các hành động của chương trình khi nó bắt đầu. Nó có thể là

■ Create new form : tạo một form mới.

■ Open last form : mở form đã được lưu trước đó.

■ Open form : mục này sẽ mở một form nào đó sau khi chương trình khởi động. Bạn có thể chọn một form nào đó có phần mở rộng là *.kxf.

- Create backup

Khi tùy chọn này được bật, chương trình sẽ tiến hành tạo một bản sao của tập tin đang làm việc (*.kxf) mỗi khi bạn lưu nó. Bản sao sẽ cùng tên với tên form, nhưng có đuôi BAK.

- Undo levels

Nút tăng giảm đơn giản này sẽ giúp bạn thiết lập số thao tác có thể Undo trong khi làm việc. Nếu bạn cấu hình nó là 0 thì sẽ vô hiệu hóa chức năng Undo.

- Number of recent file list (1-20)

Thiết lập số tập tin được được mở gần đây sẽ hiện hữu trong menu File -> Recent files.

- Association : ở đây có hai tùy chọn

■ Set : khi mở tập tin *.kxf thì nó sẽ được mở mặc định bởi chương trình Koda FormDesigner.

■ Remove : khi bạn mở tập tin *.kxf thì Windows yêu cầu bạn chọn chương trình để mở tập tin dạng này.

* Formatting

- Indent code : các tham số quy định cách thụt đầu dòng của đoạn code

■ Initial : các dòng code được tạo bởi chương trình sẽ được thụt lùi bởi số kí tự được gán tại giá trị tham số này.

■ Indent char : ở đây bạn thiết lập kí tự dùng cho thụt lùi. Có hai tham số là Tab và Space.

■ Count : ở đây bạn có thể thiết lập chiều dài của mỗi lần thụt lùi.

- Variables : thiết lập phạm vi của biến được tạo ra bởi chương trình. Nó có thể là

■ Default : không có từ khóa phạm vi của biến nào được tạo ra.

■ Global : thêm từ khóa Global cho các biến được tạo.

■ Local : thêm từ khóa Local cho các biến được tạo.

- Other

■ Data separator char : kí tự được dùng để ngăn cách các mục trong các tham số tạo control List, ListView và các control tương tự như hai control này. Mặc định sẽ là "|".

* Templates

Trong thẻ này sẽ giúp cho bạn tạo hai nút nhấn tương ứng với các template có sẵn cho loop (vòng lặp) và Events (sự kiện).

Ứng với mỗi Templates được chọn bằng cách nhấn chuột phải, bạn có thể có các lệnh sau

:

■ New : tạo mới một template.

■ Edit : Mở cửa sổ Code Template Editor để chỉnh sửa cho template được chọn. Bạn có thể mở cửa sổ này bằng cách nhấp đôi vào tên template.

■ Delete : xóa template được chọn.

■ Duplicate : tạo một bản sao cho template được chọn

■ Toggle Default : thiết lập template mặc định khi tạo code bởi chương trình. Nếu nó là template mặc định thì trường Default = Yes.

* Designer

- Grid Options

■ Display grid : hiển thị khung lưới trong cửa sổ thiết kế form

■ Snap to grid : khi một control được tạo, nó sẽ tự động canh lề đến dòng lưới gần nhất.

■ Grid size : kích thước chiều ngang và chiều dọc của lưới. Mặc định là 8 X 8 pixel.

- Size and position saving : các tham số quy định vị trí và kích thước của cửa sổ thiết kế. Nó có thể là :

■ Don't save : cửa sổ thiết kế sẽ được bắt đầu tại vị trí và kích thước mặc định.

■ Automatically : Koda sẽ tự động nhớ vị trí và kích thước của cửa sổ thiết kế.

■ Manually : kích thước và vị trí của cửa sổ thiết kế sẽ được lưu bởi người dùng. (bạn có thể vào thực đơn Options -> Remember position).

■ Keep standard layout : khi di chuyển cửa sổ, Koda FormDesigner sẽ duy trì giao diện chuẩn của nó.

- Object Tree

■ Always full expand : khi tùy chọn này được bật thì cây trong Object TreeView sẽ luôn được bung đầy đủ ra.

■ By default use Caption instead Name : dùng thuộc tính Caption thay vì thuộc tính Name của control để dùng hiển thị trong cửa sổ Object TreeView.

* *Color*

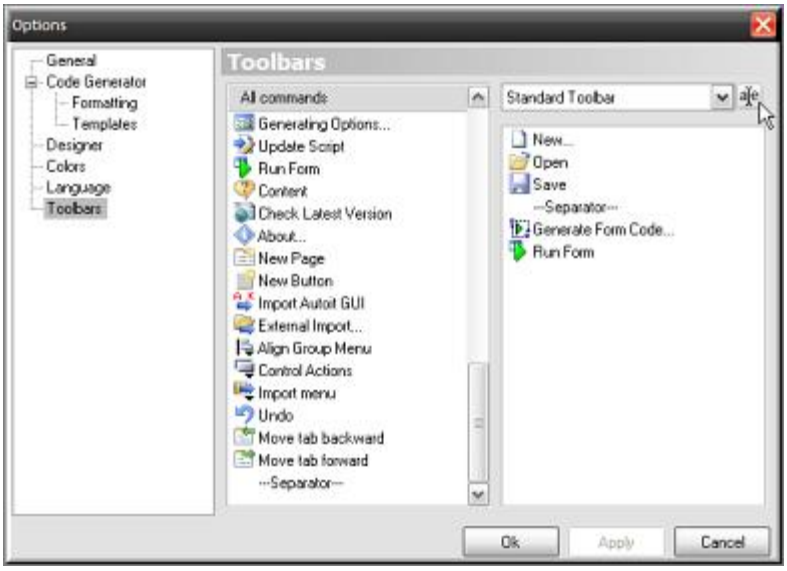
Trong nhóm này, bạn có thể thay đổi màu sắc cho các phần tử của một nhóm nào đó. Bao gồm ba nhóm là Designer, Object Inspector, Output Window. Ứng với mỗi nhóm này, bạn có thể thay đổi màu sắc các thành phần trên mỗi nhóm.

* *Language*

Thay đổi ngôn ngữ hiển thị cho chương trình.

* *Toolbars*

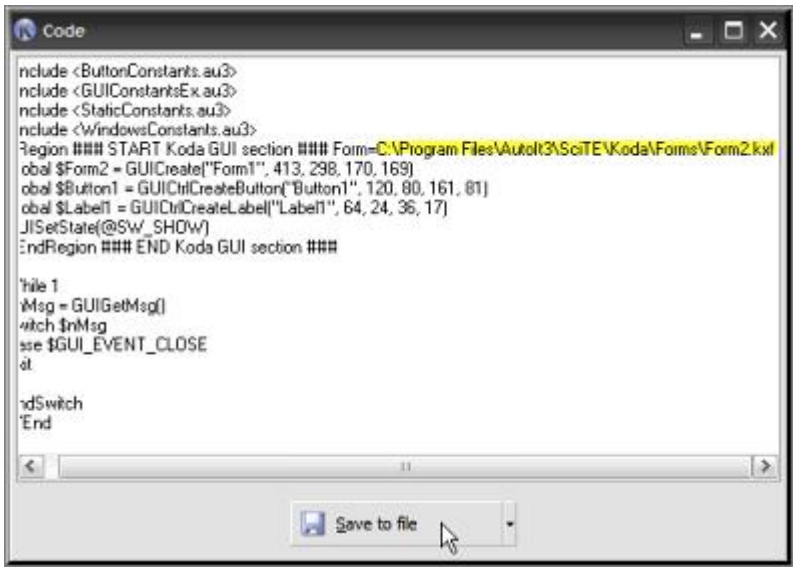
Tùy biến thanh công cụ của chương trình. Chương trình có 5 thanh công cụ chính đó là : Standard Toolbar, Function, User toolbar 1, User toolbar 2, User toolbar 3. Mặc định thì chỉ có Standard Toolbar và Function là có nút lệnh, riêng ba thanh công cụ còn lại không có nút lệnh nào cả. Để thêm một nút lệnh mới cho thanh công cụ nào đó, bạn chỉ cần rê thả nút lệnh cần thêm vào thanh công cụ đó. Để gỡ bỏ nút lệnh đã có trên thanh công cụ, hãy làm ngược lại. Để sửa tên một thanh công cụ, bạn có thể nhấn biểu tượng có chữ a và e, ở giữa là kí tự con nháy gõ văn bản. Trong các nút lệnh nhóm All commands, có lệnh –Separator--, đó là dấu ngăn cách đứng giữa các nhóm lệnh trong một thanh công cụ.



5. Tools

Trong trình đơn Tools có các lệnh Generate code (F9) : xuất hiện cửa sổ Code Dialog chứa kết quả code được tạo ra tương ứng với giao diện form mà bạn đã thiết kế. Run Form (F10) : xem thử giao diện mà bạn đã tạo. Generating Options (Ctrl – F9) : mở hộp thoại Generating Options dùng để thiết lập một số tham số tùy chọn cho code được tạo ra bởi chương trình như thiết lập code OnEvent, thiết lập thụt đầu dòng cho code xuất ra, tạo các sự kiện mặc định cho control.

Trong menu Tools có lệnh Update Script (Ctrl – U) : công dụng của lệnh này có thể hiểu đơn giản như thế này. Mỗi form bạn tạo trong chương trình sẽ được lưu lại đến dạng tập tin *.kxf. Trong code chương trình tạo ra cho form thì sẽ có đường dẫn đến tập tin của form này. Trong trường hợp bạn xuất đoạn code của chương trình tạo ra thành dạng tập tin *.au3 (Tools -> Generate Form Code, chọn phương thức Save to file) thì sau khi bạn lưu xong, nếu bạn có cập nhật gì giao diện của mình thì bạn hãy chọn lệnh Tools -> Update Script để chương trình cập nhật lại tập tin *.au3 tương ứng với sự thay đổi của form sau khi lưu.



III. Tìm hiểu các control và thuộc tính của chúng

1. Forms

Forms nói chung là một cửa sổ giao diện chứa các control khác. Nó được tạo khi chọn

File -> New hoặc bạn nhấn nút New trên thanh công cụ.

Bảng thuộc tính của form

Caption	Nội dung văn bản sẽ hiển thị trên thanh tiêu đề của form.
ClientHeight	Độ dài chiều cao của khu vực bên trong control (không tính thanh tiêu đề và đường biên dưới).
ClientWidth	Độ dài chiều ngang của khu vực bên trong control (không tính độ dài của hai đường biên trái và phải).
Color	Thuộc tính quy định màu nền trong control.
ContexMenu	Cho phép đính kèm control ContextMenu (thực đơn xuất hiện khi nhấn chuột phải trên form) đến form. Tuy nhiên control ContextMenu cần phải được tạo trước tiên.
Cursor	Thuộc tính định nghĩa kiểu con trỏ chuột xuất hiện khi người dùng di chuyển chuột lên control.
Description	Thông tin mô tả về form.
Enabled	Mặc định là True. Nếu bạn thiết lập nó thành False thì sẽ vô hiệu hóa control.
Font	Thuộc tính cho phép bạn thiết lập thuộc tính font chữ hiển thị như màu sắc, kích cỡ, in đậm, in nghiêng, gạch dưới, gạch ngang chữ.
Left, Top	Vị trí của control (tọa độ trái cửa sổ, tọa độ đỉnh trên của cửa sổ).
Width, Height	Độ dài chiều ngang, chiều rộng của control (tính bằng pixel).
Hint	Văn bản sẽ xuất hiện khi con trỏ chuột di chuyển trên control.
Icon	Chỉ định một tập tin biểu tượng cho form. Nhấn nút … sẽ mở hộp thoại Picture Editor để bạn chọn tập tin làm icon.
Menu	Thuộc tính chỉ định danh menu cần gán cho form. Control Menu phải được tạo đầu tiên.
Name	Tên biến sẽ chứa định danh ID của hàm tạo control form. Nếu bạn để tham số này rỗng, sẽ không có biến cho control form tạo ra trong code kết quả.
ParentForm	Tên của cửa sổ cha cho cửa sổ hiện tại. Thường dùng khi tạo cửa sổ con.
Position	Tham số quy định vị trí của form khi chạy chương trình. poDesigned – vị trí của form sẽ như vị trí khi bạn thiết kế. poDesktopCenter – vị trí của form sẽ nằm giữa màn hình. poFixed – vị trí của form sẽ nằm ở tọa độ (0, 0) của màn hình, tính theo tọa độ của đỉnh trên bên trái của form.

TrayMenu	Cho phép đính kèm control TrayMenu đến form. TrayMenu là một loại menu khi người dùng nhấn chuột vào biểu tượng icon của chương trình trên khay hệ thống. Để sử dụng thuộc tính này, thì control TrayMenu cần phải được tạo trước tiên.
Visible	Mặc định là True. Nếu bạn gán nó là False thì control sẽ bị ẩn đi.

2. Menu

Tạo một thanh thực đơn trên form. Trong cửa sổ thuộc tính, bạn chú ý đến Items. Đây là mục để bạn xây dựng thanh thực đơn của bạn. Bạn hãy nhấn vào dấu … để mở cửa sổ Menu Designer để thiết kế thanh thực đơn của mình.

3. Label

Tạo một điều khiển Label trên form.

Bảng thuộc tính của label

Align	Thuộc tính quy định canh biên đối tượng label. Tuy nhiên canh biên này là dạng canh biên so với form. Ví dụ, nếu bạn quy định tham số này là alBottom thì Label sẽ nằm dưới đáy form.
AutoSize	Nếu tham số này là True, độ dài chiều rộng của label sẽ tự động điều chỉnh cho phù hợp với độ rộng của văn bản trong label.
TabOrder	Thuộc tính định nghĩa thứ tự của control được tạo. Đó là thứ tự mà con nháy đi đến control nào đó khi bạn nhấn phím Tab. Bạn có thể dùng cửa sổ Tab Order Editor để chỉnh sửa thứ tự này.

4. Input

Tạo một control trên form dùng để nhập thông tin, tuy nhiên chỉ nhập trên một dòng.

Bảng thuộc tính của Input

MaxLength	Số kí tự tối đa mà người dùng có thể nhập liệu trên control.
Text	Thuộc tính chỉ định văn bản sẽ hiển thị trên control.
UpDown	Gán một control UpDown (nút tăng giảm giá trị) đến control Input. Tuy nhiên bạn phải tạo control UpDown trước khi có thể sử dụng thuộc tính này.

5. Edit

Tạo một control trên form dùng để nhập thông tin. Tuy nhiên điểm khác giữa Edit và Input là bạn có thể nhập thông tin trên nhiều dòng. Trong cửa sổ thuộc tính, bạn chú ý đến mục Lines. Bạn hãy tiến hành nhấn vào dấu … để mở cửa sổ String List Editor để nhập thông tin văn bản vào control.

6. Button

Tạo một nút nhấn trên form.

7. Checkbox

Tạo một hộp kiểm checkbox để cho người dùng nhấn chọn. Trong cửa sổ thuộc tính, bạn hãy chú ý đến Checked. Nếu tham số này là True, Checkbox sẽ được chọn mặc định. Nếu tham số này là False, Checkbox sẽ ở trạng thái không được chọn.

8. Radio

Tạo một nút nhấn Radio, nó tương tự như Checkbox tuy nhiên nó có dạng hình tròn.

9. ListBox

Tạo một control chứa danh sách. Danh sách này chỉ có một cột nhưng có nhiều dòng. Trong cửa sổ thuộc tính có mục Items. Bạn hãy tiến hành nhấn dấu … mở cửa sổ String List Editor để tiến hành nhập liệu dữ liệu các mục trên control này.

10. ComboBox

Tạo một hộp Combobox chứa danh sách đổ xuống để lựa chọn một mục nào đó.

Bảng thuộc tính của ComboBox

Items	Nhấn vào dấu … để mở cửa sổ String List Editor để giúp cho bạn nhập các mục trong ComboBox.
ItemsIndex	Số index của mục mặc định sẽ được chọn trên ComboBox, bắt đầu từ 1. Nếu tham số này là 2, mục thứ 2 trong ComboBox sẽ được chọn làm mặc định.

11. ContextMenu

Tạo một menu chuột phải trên form. Menu này có thể gán cho một đối tượng form hay một control nào đó. Bạn hãy nhấn vào dấu … ngay thuộc tính Items để mở cửa sổ Menu Designer để thiết kế menu này.

12. Group

Tạo một control Group, dùng để nhóm các đối tượng lại với nhau (thường áp dụng cho Checkbox hay Radio button).

13. Picture

Tạo một control dùng để hiển thị ảnh trên form.

Bảng thuộc tính của Picture

Picture	Thuộc tính chỉ định ảnh sẽ hiển thị trên control. Bạn nhấn dấu … để hiển thị cửa sổ Picture Editor để bạn chọn ảnh.
Stretch	Thuộc tính chỉ định cách căng ảnh trong khi thay đổi kích thước ảnh trong thiết kế. Nếu thuộc tính này là sFree, bạn có thể resize ảnh theo chiều rộng hoặc theo chiều ngang cũng được. Nếu thuộc tính này là sProportional thì khi resize ảnh, ảnh sẽ tự động chỉnh lại kích thước theo tỉ lệ gốc của ảnh. Nếu tham số này là sRealSize thì bạn không thể resize ảnh.

14. Icon

Tạo một control dùng để hiển thị một icon.

15. Graphic

Tạo một control dùng để hiển thị các đối tượng trong hình học.

Bảng thuộc tính cho Graphic

BgColor	Thuộc tính chỉ định màu nền cho control
Items	Nhấn vào nút … để mở cửa sổ Graphic Editor giúp bạn chèn một số đối tượng hình học vào control.

16. UpDown

Tạo một control có hai nút để tăng giảm giá trị. Control này thường đi kèm với control Input.

Bảng thuộc tính cho UpDown

Max	Giá trị tối đa mà người dùng có thể điều chỉnh.
Min	Giá trị tối thiểu mà người dùng có thể điều chỉnh.

17. Avi

Tạo một điều khiển control dùng để hiển thị tập tin video có phần mở rộng là *.avi

Bảng thuộc tính của Avi

CommonAVI	Đây là thuộc tính cung cấp cho bạn một số đoạn video AVI phổ biến thường được dùng như đoạn video diễn cảnh đang sao chép tập tin (aviCopyFile), đang xóa tập tin (aviDeleteFile) thay vì dùng tập tin do người dùng cung cấp,…
FileName	Đường dẫn của tập tin AVI cần hiển thị.

Control tạo ra trong chương trình thiết kế chỉ có dạng tĩnh (không trình diễn video khi chạy thử). Nếu bạn muốn video trình diễn thì trong đoạn code tạo ra bởi chương trình, bạn thêm dòng lệnh GUICtrlSetState($avi, 1) – để chơi tập tin hoặc GUICtrlSetState($avi, 0) - để ngừng chơi tập tin. Trong đó tham số $avi là biến chứa chỉ số ID của hàm tạo control trong đoạn code tạo Avi.

18. TrayMenu

ươi Control này dùng để tạo một menu tắt khi nhấn chuột vào biểu tượng của chương trình chạy trên khay hệ thống.

Bảng thuộc tính của TrayMenu

Items	Nhấn dấu … để mở cửa sổ Menu Designer. Cửa sổ này giúp bạn tạo các mục trong menu.
MouseClick	Thuộc tính quy định menu tắt này sẽ hiển thị khi người dùng nhấn chuột như thế nào. mcPriPress (khi chuột trái

20. Slider

Tạo một thanh trượt theo kiểu di chuyển sang trái phải để thiết lập giá trị. Các thuộc tính của nó cũng tương tự như control UpDown.

21. Progress

Tạo một control dùng để hiển thị thanh tiến trình tương tự như thanh tiến trình bạn thấy khi tải một trang web nào đó trong trình duyệt Internet Explorer.

22. Date Picker

Tạo một control dùng để cho người dùng chọn ngày.

Bảng thuộc tính của Date Picker

Date	Giá trị ngày được thiết lập mặc định trên control. Định dạng ngày gõ vào bạn phải tuân thủ theo định dạng mà bạn đã thiết lập trong Windows (Start -> Control Panel -> Regional and Language Options).
Format	Thiết lập định dạng ngày giờ sẽ hiển thị trên control. dd : ngày, MM : tháng, yyyy : năm, hh : giờ, mm : phút, ss : giây. Ví dụ dd/MM/yyyy sẽ hiển thị ngày hôm nay của đồng hồ hệ thống trên máy tôi là 29/07/2011.
Time	Giá trị thời gian (giờ, phút, giây) được thiết lập mặc định trên control. Định dạng thời gian gõ vào bạn phải tuân thủ theo định dạng mà bạn đã thiết lập trong Windows (Start -> Control Panel -> Regional and Language Options).

23. Month Calendar

Tạo một control dùng để hiển thị lịch theo tháng.

Bảng thuộc tính của Month Calendar

Autosize	Nếu tham số này là True thì control sẽ không thể thay đổi kích thước ngay khi được tạo.
Date	Giá trị ngày mặc định sẽ hiển thị trên control. Định dạng ngày gõ vào bạn phải tuân thủ theo định dạng mà bạn đã thiết lập trong Windows (Start -> Control Panel -> Regional and Language Options).

24. TreeView

Tạo một control dùng để hiển thị các mục theo dạng cây. Ví dụ điển hình thường thấy cho TreeView là bạn hay duyệt cây thư mục trong Windows Explorer.

Bảng thuộc tính cho TreeView

Images	Đính kèm control ImageList vào TreeView nhằm mục đích lấy hình ảnh trong ImageList làm biểu tượng các mục trong TreeView.
Items	Bạn nhấn dấu … để mở cửa sổ Treeview Items Editor giúp bạn tạo các nút trong TreeView cũng như chọn ảnh từ control ImageList làm ảnh biểu tượng cho các nút.

25. ListView

Tạo một control dùng để hiển thị nội dung tương tự như control ListBox, tuy nhiên nó có

nhiều cột.

Bảng thuộc tính cho ListView

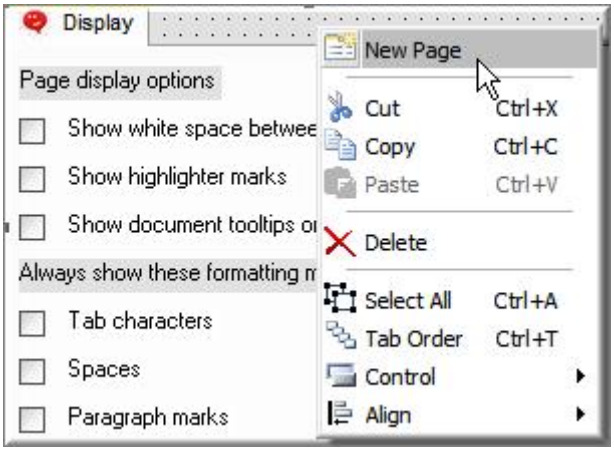
Columns	Nhấn dấu … để mở cửa sổ Collection Editor. Trong hộp thoại này sẽ giúp bạn tạo các tiêu đề cột cho control này.
Items	Nhấn dấu … để mở cửa sổ ListView Items Editor giúp bạn tạo các mục trong control này cũng như chọn ảnh đại diện.

26. Tab Control

Tạo control Tab dùng để hiển thị các nội dung cũng như control theo dạng thẻ. Để tạo một thẻ mới cho control, bạn nhấn chuột phải lên control và chọn New Page. Ứng với mỗi thẻ được tạo ra, bạn có thể bung mở rộng tại thuộc tính ActivePage để gán một số thuộc tính cho thẻ hiện thời như : Caption (tiêu đề thẻ), ImageIndex (chỉ mục ảnh trong ImageList dùng làm ảnh đại diện cho thẻ, tính từ 0).

Bảng thuộc tính cho control

ActivePage	Thẻ sẽ hiển thị mặc định khi chương trình chạy.
Images	Đính kèm control ImageList. Khi đính kèm control này vào Tab sẽ giúp cho phép bạn chọn ảnh đại diện cho từng thẻ.



27. Status Bar

Tạo một thanh trạng thái tương tự như thanh trạng thái trong chương trình soạn thảo văn bản như Microsoft Word.

Bảng thuộc tính cho Status Bar

Panels	Nhấn vào dấu … để mở cửa sổ Collection Editor giúp bạn tạo tiêu đề cho từng panel sẽ hiển thị dưới thanh trạng thái. Mỗi panel sẽ được ngăn cách bởi một dấu sổ đứng.
SimplePanel	Nếu tham số này là True thì thanh trạng thái của bạn chỉ có một panel duy nhất. Đây là dạng hiển thị đơn giản nhất của control này.
SimpleText	Thuộc tính văn bản sẽ hiển thị trên thanh trạng thái trong chế độ SimplePanel.

28. IP Adress InputBox

Tạo một control dùng để nhập địa chỉ IP. Trong bảng thuộc tính, bạn chú ý đến thuộc

tính Text. Bạn gõ địa chỉ IP tại mục này dùng để làm IP mặc định khi chương trình chạy.

29. Toolbar

Tạo một thanh công cụ trên form. Trong bảng thuộc tính, bạn chú ý đến thuộc tính Image. Trong thuộc tính này, bạn chỉ định control ImageList tạo trước đó để dùng những ảnh có trong control này làm ảnh biểu tượng cho các nút lệnh trên thanh công cụ.

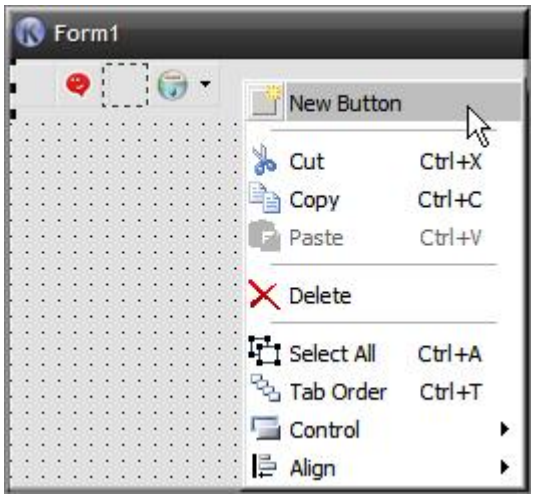
Sau khi bạn đặt control Toolbar lên form đang thiết kế, bạn có thể tạo một hoặc nhiều nút công cụ. Để làm điều đó, bạn có thể nhấn phải chuột vào control sau đó chọn New Button. Một nút mới sẽ hiện ra sau đó.

Khi bạn làm việc với control này hãy lưu ý là có sự khác nhau giữa thanh công cụ được chọn và nút trên thanh công cụ được chọn.

Để xóa một nút, bạn có thể nhấn phải chuột vào nó sau đó chọn Delete hoặc nhấn phím Delete trên bàn phím.

Mỗi nút trên thanh công cụ sẽ những thuộc tính. Nhưng bạn hãy chú ý đến thuộc tính style, đây là thuộc tính chỉ định cho loại nút được tạo ra. tbsButton : nút nhấn thông thường, tbsCheck : nút nhấn dạng check, tbsDropdown : nút nhấn với thanh menu đổ xuống tương tự như Combobox, tbsSeparator : nút sổ đứng dùng để ngăn cách các nút nhấn trên thanh công cụ với nhau.

Điểm hạn chế của cách tạo này chỉ là tạo giao diện bề ngoài mà không có tạo chi tiết bên trong. Chẳng hạng như nút nhấn có style tbsDropDown thì không thể thêm bớt các mục nằm trong nút nhấn đó.



30. ImageList

Tạo control dùng để chứa ảnh, thường được dùng làm ảnh đại diện cho một số control được nhắc đến trên đây.

Bảng thuộc tính cho ImageList

ColorDepth	Độ sâu màu cho control. Mặc định là 32 bit.
ImageWidth, ImageHeight	Độ dài chiều rộng, chiều cao của các ảnh trong control.
Images	Nhấn dấu … để mở cửa sổ ImageList Editor để cho bạn chọn các ảnh trong control này.

31. COM Object

Tạo một control chứa các điều khiển ActiveX.

Bảng thuộc tính cho Com Object

AXName	Tên cho control này.
AXObject	Mở cửa sổ ActiveX objects browser để chọn thành phần ActiveX.

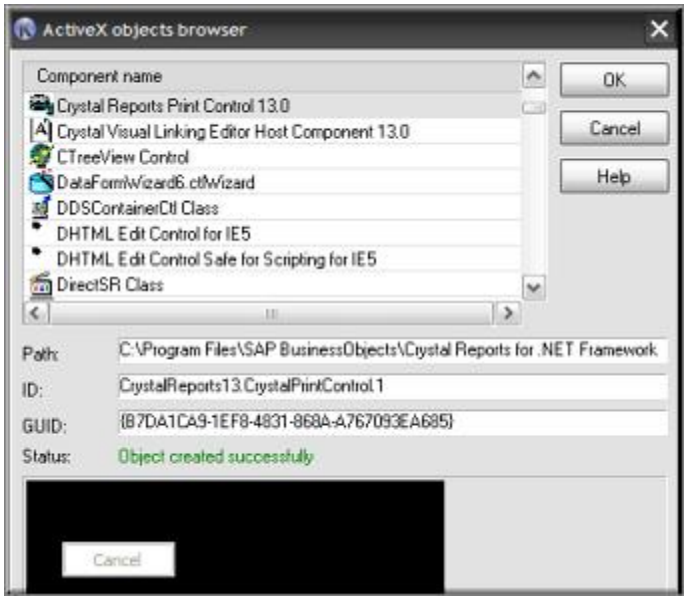
IV. Tìm hiểu các cửa sổ/hộp thoại khi thiết lập thuộc tính cho control

1. ActiveX objects browser

Cửa sổ này sẽ xuất hiện khi bạn chọn thuộc tính AX Object của control COM Object để bạn chọn thành phần ActiveX muốn dùng. Tuy nhiên bạn phải chú ý là phải save form của bạn lại trước khi chọn đối tượng ActiveX. Mặc dù Koda FormDesigner sẽ cố gắng xử lý lỗi xảy ra bởi những thành phần này, tuy nhiên khó mà lường trước được, một số thành phần ActiveX có thể gây lỗi cho chương trình hoặc đóng băng và do đó bạn sẽ không kịp lưu lại những form đang thiết kế của mình.

Cửa sổ ActiveX objects browser có thể tốn một chút thời gian để mở do phải scan các thành phần ActiveX được cài đặt trong Registry.

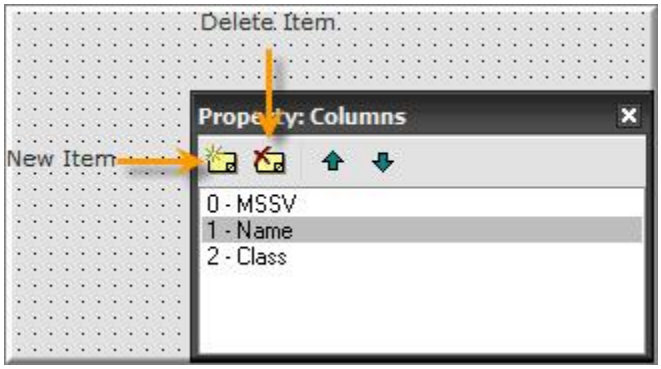
Khi bạn chọn một thành phần ActiveX trong danh sách, bạn có thể thấy một số thông tin như Path, ID, GUID, và cửa sổ xem thử đối tượng ActiveX đã tạo ngay khung Preview. Bạn có thể kiểm tra thông tin ngay dòng Status để biết nó có được tạo thành công hay không.



2. Collection Editor

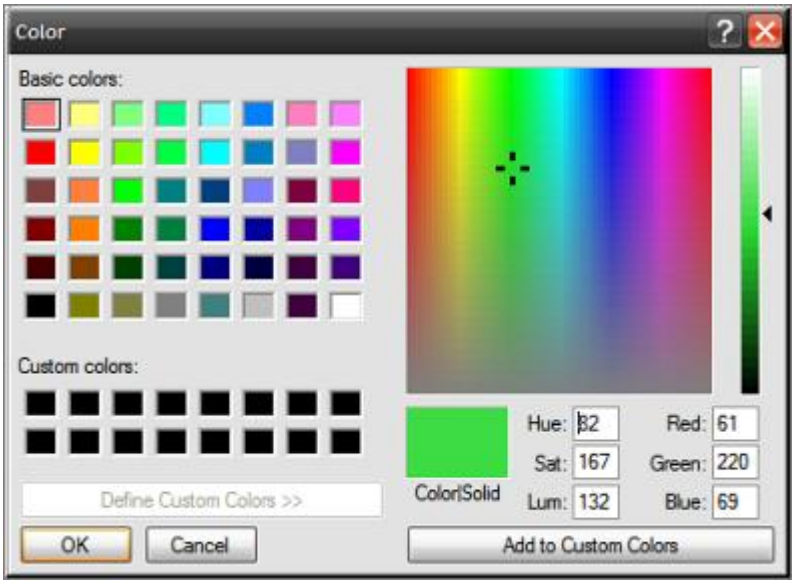
Cửa sổ này sẽ xuất hiện khi bạn chọn thuộc tính Columns trong ListView. Cửa sổ này sẽ giúp thêm tiêu đề cột cho ListView.

Để thêm một mục mới bạn có thể nhấn Add Item. Bạn có thể thay đổi thuộc tính cho mỗi trong cửa sổ Object Inspector. Đừng quên nhấn Enter sau khi bạn đã thay đổi thuộc tính. Để xóa một mục nhấn Delete Item. Bạn có thể sắp xếp lại vị trí các mục bằng cách nhấn hai nút ↓, ↑. Sau khi tạo xong các mục trong cửa sổ này, bạn chỉ cần đóng hộp thoại là được.



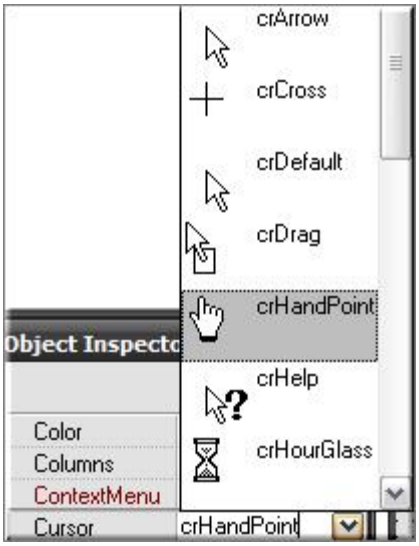
3. Color Editor

Color là thuộc tính có hầu hết trên các control giúp bạn tạo màu cho control nào đó. Bạn có thể chọn màu từ danh sách đổ xuống theo kiểu combobox từ thuộc tính Color của đối tượng. Hoặc có thể chọn màu thông qua hộp thoại Color bằng cách nhấn đôi chuột vào thuộc tính này. Bạn có thể thiết lập màu trực tiếp thông qua thuộc tính Color bằng cách gõ vào $00RRGGBB. Trong đó RRGGBB là mã màu dạng hexa. (Red – Green – Blue)



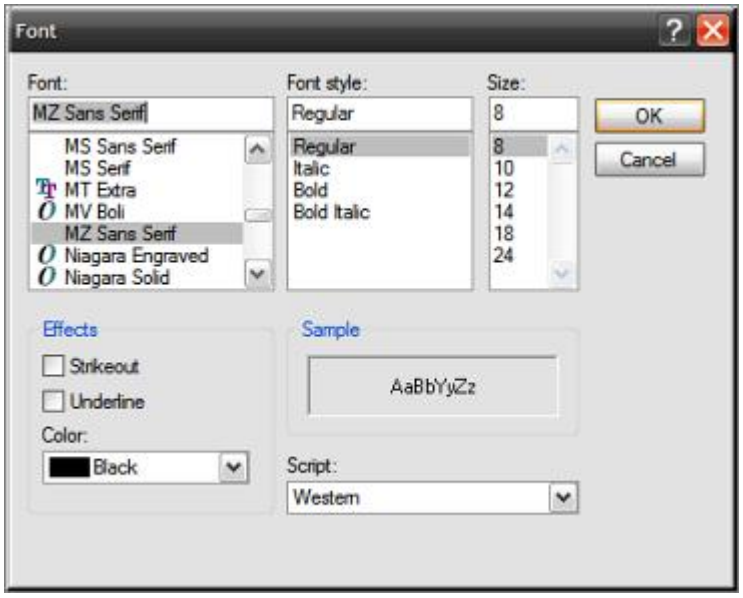
4. Cursor Editor

Trong một số control có thuộc tính Cursor để cho phép bạn chọn kiểu chuột sẽ xuất hiện khi di chuyển qua control. Không có gì đặc biệt cho thuộc tính này. Đơn giản bạn chỉ chọn cho mình một con trỏ chuột mà mình thích mà thôi.



5. Font Editor

Hộp thoại này sẽ xuất hiện khi bạn xác lập thuộc tính Font cho một số control có hỗ trợ chẳng hạn như Label, Button,… Khi bạn nhấn nút … trong thuộc tính Font sẽ mở một hộp thoại Windows chuẩn rất quen thuộc cho phép bạn thiết lập một số tùy chọn cho văn bản.

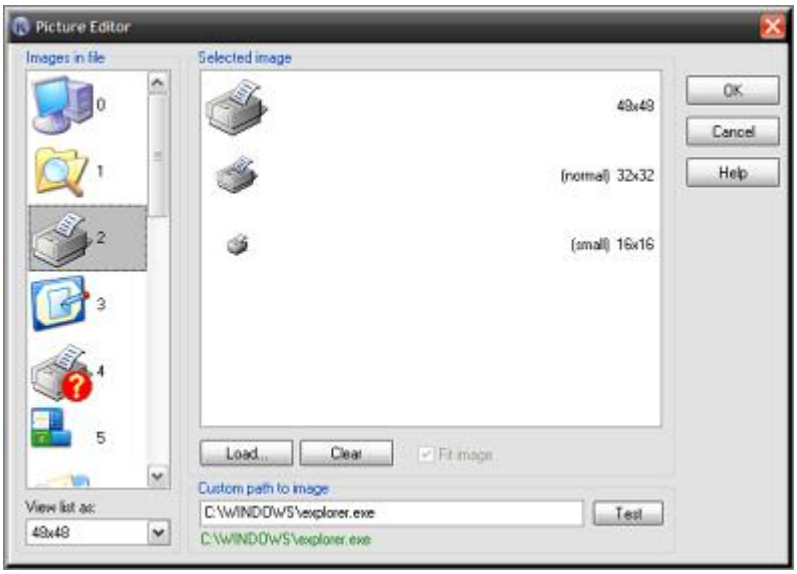


6. Picture Editor

Hộp thoại này xuất hiện khi bạn thiết lập thuộc tính icon cho Form, thuộc tính Picture của Icon, thuộc tính Picture của control Picture, thuộc tính icon của Tray Menu.

Hộp thoại Picture Editor được chia thành hai phần. Phần bên trái sẽ chứa danh sách các ảnh của một tập tin mà người dùng chọn. Nó có thể là hình ảnh trong một tập tin exe, dll, ocx. Ở phía dưới sẽ có một combobox View list as cho phép bạn chọn chế độ hiển thị theo kích thước cho sẵn. Ở khung bên phải cho phép hiển thị các icon hoặc hình ảnh được chọn. Nếu ảnh bạn chọn là một icon, khung bên phải này sẽ chứa danh sách các icon từ lớn đến nhỏ để cho phép bạn có thể lựa chọn. Trong trường hợp ảnh bạn chọn là một tập tin ảnh thì khung bên phải này sẽ hiển thị hình ảnh được chọn. Bạn có thể đánh dấu check vào hộp kiểm Fit Image để hiển thị hình ảnh đúng kích thước gốc của nó. Bạn hãy nhấn nút Load để chọn tập tin ảnh cần dùng hoặc nhấn vào nút Clear để xóa ảnh. Bạn có thể chọn ảnh bằng tay trong trường hợp bạn biết đường dẫn bằng cách gõ đường

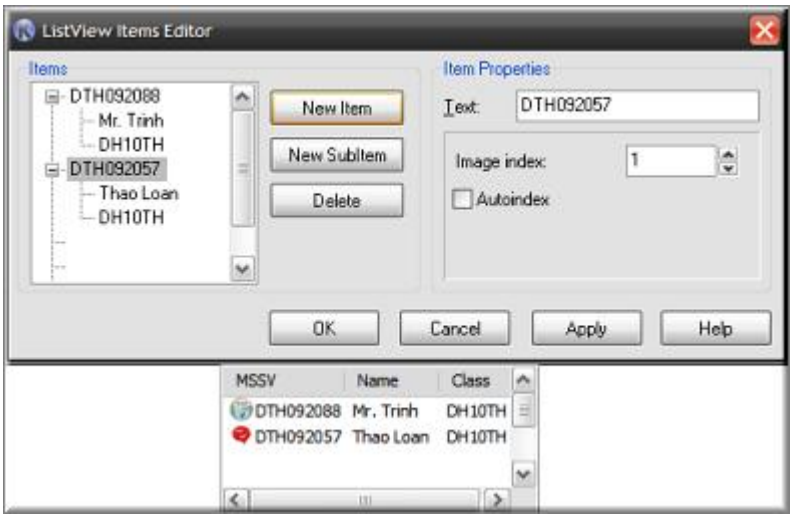
dẫn đó trong Custom path to image.



7. ListView Items Editor

Cửa sổ này tương tự như cửa sổ của TreeView Items Editor, hộp thoại này sẽ được mở khi bạn thiết lập thuộc tính Items của control ListView. Tuy nhiên bạn chỉ tạo các mục cho control ListView hai cấp mà thôi. Cấp đầu trong cây sẽ là nội dung mục cho cột thứ nhất, các mục con còn lại trong cấp đầu sẽ là nội dung các mục trong các cột còn lại.

Bạn hãy nhấn nút New Item để tạo một mục mới trong ListView. Nhấn vào nút New Subitem tạo một mục con dưới một mục được chọn. Nhấn nút Delete để xóa một mục hoặc mục con. Ở khung bên phải dùng để cho bạn thay đổi thuộc tính các mục trong ListView. Text là nội dung văn bản cho một mục được chọn. Image index là chỉ mục hình ảnh cần làm ảnh đại diện cho các mục trong ListView (bạn cần phải tạo control ImageList để chứa ảnh. Mỗi ảnh trong control ImageList sẽ gán một chỉ số index, ảnh đầu tiên sẽ gán là 0, ảnh tiếp theo gán là 1,…). Tùy chọn Autoindex cho phép tự động tăng chỉ số ảnh làm ảnh đại diện cho mục khi bạn tạo một mục mới trong ListView. Nội dung văn bản trong mỗi mục sẽ thay đổi ngay lập tức trong khi bạn gõ văn bản trong trường Text. Để hiểu rõ hơn, bạn hãy xem hình ảnh sau đây.



8. TreeView Items Editor

Cửa sổ này được mở khi bạn thiết lập thuộc tính Items cho control TreeView. Cửa sổ này có các nút và thuộc tính chỉnh sửa tương tự như ListView Items Editor. Điểm khác biệt giữa hai cửa sổ này là TreeView Items Editor cho phép bạn tạo cây chứa các mục nhiều

cấp chứ không phải hạn chế hai cấp như ở ListView Items Editor.

9. Menu Designer

Cửa sổ này sẽ được mở khi bạn thiết lập thuộc tính Items cho các control như Main Menu, Context Menu và Tray Menu. Mục đích của cửa sổ này nhằm giúp cho người dùng thiết kế menu. Lần đầu tiên bạn mở cửa sổ này, bạn sẽ thấy một khung giữ chỗ màu xám hoặc màu trắng. Đó là nơi cho bạn biết vị trí của mục mới sẽ được tạo ngay tại vị trí đó. Nhấn chuột phải tại khung giữ chỗ đó, sẽ có một menu tắt xuất hiện giúp bạn thực hiện một số thao tác như Insert (thêm mục mới), Delete (xóa mục được chọn). Bạn cũng có thể tạo một menu con cho mục đang chọn bằng cách chọn vào Create Submenu.

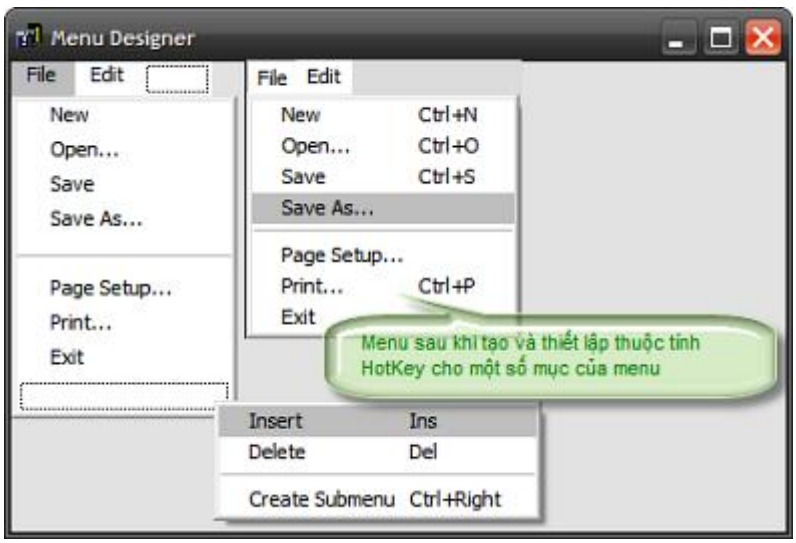
Đối với mỗi mục của menu, bạn có thể thiết lập một số thuộc tính như : Checked (nếu tham số này là true, mục chọn menu sẽ có một dấu check bên trái nó), HotKey (thiết lập tổ hợp phím nóng cho mục đang chọn).

Chú ý :

- Bạn có thể dùng chuột để thực hiện thao tác rê và thả các mục đến vị trí cần thiết và cũng để sắp xếp lại thứ tự các mục trong menu.

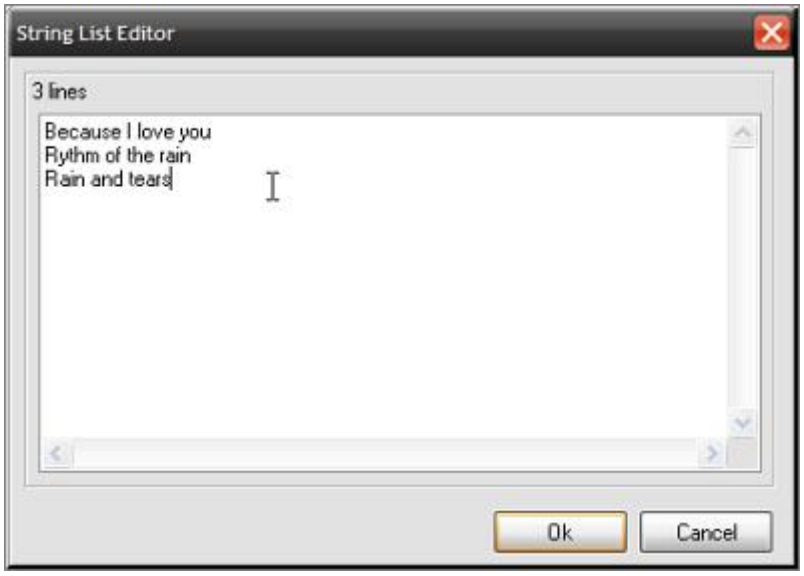
- Nếu bạn tạo một menu con trong một mục của menu mà không tạo mục con trong nó, nó sẽ trở lại thành một mục bình thường ngay sau khi bạn mở cửa sổ Menu Designer.

- Để ngăn cách các lệnh trong một menu bằng dấu gạch ngang, bạn có thể tạo một mục menu mới sau đó trong thuộc tính Caption bạn gán là "-" (dấu -).



10. String List Editor

Cửa sổ này sẽ xuất hiện khi bạn thiết lập thuộc tính Lines cho control Edit. Không có gì đặc biệt cho cửa sổ này. Nó là một cửa sổ cho phép bạn nhập dữ liệu văn bản nhiều dòng cho control.

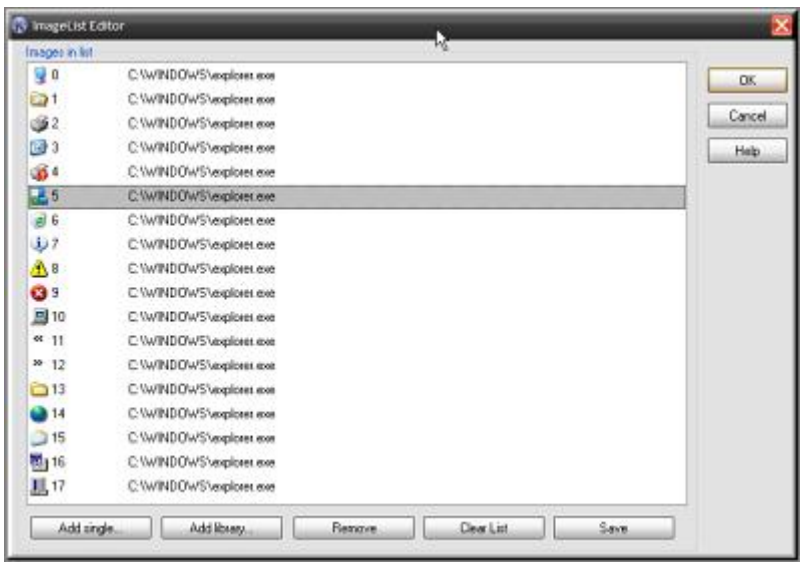


11. ImageList Editor

Cửa sổ này sẽ xuất hiện khi bạn thiết lập thuộc tính Images của control ImageList. Cửa sổ này sẽ giúp bạn thêm các ảnh vào control này. Trong cửa sổ này có các nút như sau : Add single : chỉ thêm một tập tin ảnh, kể cả khi bạn thêm một tập tin có hỗ trợ nhiều hình ảnh như *.exe, *.dll, Add library : thêm một tập tin mà nó có hỗ trợ nhiều ảnh như tập tin *.exe hoặc *.dll, Remove : gỡ bỏ ảnh được chọn, Clear List : xóa tất cả các ảnh được chọn, Save : lưu trữ tất cả ảnh trong danh sách thành một tập tin *.dll.

Ghi chú :

Các hệ điều hành mới hơn như Windows Vista/7 không hỗ trợ các tập tin icon dạng 16-bit (*.icl, *.nil thường là 16-bit). Koda FormDesginer có thể hỗ trợ lưu các tập tin ảnh dạng 16–bit được chọn thành dạng 32-bit để có thể hỗ trợ bởi các hệ điều hành mới hơn. Vì thế, trên Windows Vista/7, bạn có thể tải các tập tin chứa ảnh dưới dạng 16-bit. Sau đó bạn hãy lưu nó lại thành tập tin thư viện *.dll 32-bit bằng chức năng Save của cửa sổ ImageList Editor để có thể sử dụng được trên Windows Vista/7.



12. Graphic Editor

Cửa sổ này sẽ xuất hiện khi bạn thiết lập thuộc tính Items cho control Graphic. Cửa sổ này sẽ giúp bạn tạo các ảnh hình học cho control Graphic. Khu vực thiết kế của cửa sổ sẽ được đóng khung với một đường lưới giúp cho bạn thiết kế dễ dàng hơn, có thể bật tắt

lưới này bằng cách chọn Options -> Show grid. Ngay tại khung bên phải sẽ chứa danh sách các hình được sắp xếp theo thứ tự được tạo. Ở phía dưới danh sách sẽ là cửa sổ thuộc tính mà bạn có thể thay đổi cho đối tượng hình học được chọn. Ở phía dưới ngay thanh trạng thái của chương trình, bạn có thể thấy tọa độ con trỏ hiện thời trên màn hình (Point) và số đối tượng hình học hiện có trên khu vực thiết kế (Objects).

Việc chèn các đối tượng hình học cũng tương tự như control. Bạn hãy chọn hình mà mình muốn chèn trên thanh công cụ, sau đó dùng chuột kéo thả và chỉnh sửa kích thước của hình sau cho phù hợp rồi nhả chuột ra.

Khi một đối tượng hình được chèn, bạn có thể thực hiện các thao tác đối với đối tượng được chèn như sau :

- Thay đổi kích thước của đối tượng hình học bằng cách rê thả các điểm chọn trên đối tượng. (điểm chọn có dạng hình vuông nhỏ)

- Di chuyển các đối tượng hình học bằng cách dùng chuột rê và thả nó đến vị trí mong muốn.

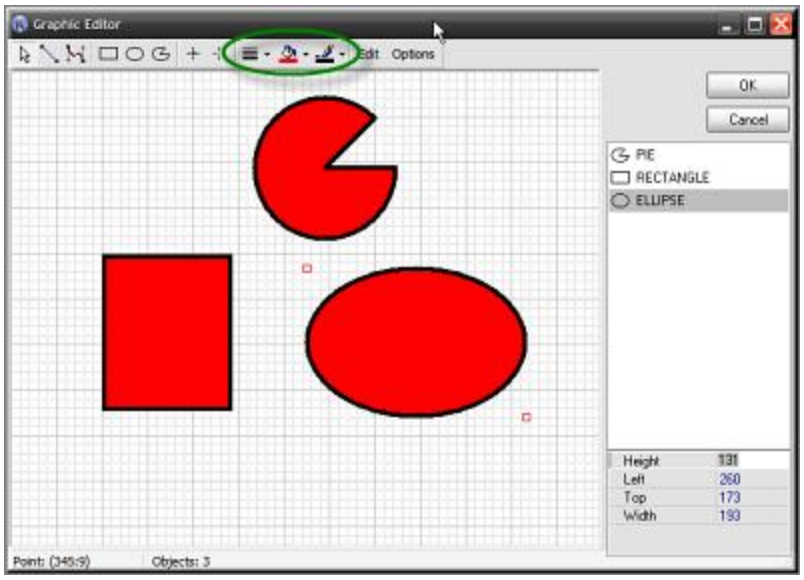
- Thay đổi thuộc tính của đối tượng hình trong cửa sổ thuộc tính. Đối với đối tượng hình Line và Bezier, bạn có thêm thuộc tính ShowPoint. Nếu bạn thiết lập thuộc tính này là True, thì hai đối tượng trên sẽ có thêm một dấu + ngay tại hai đầu của chúng.

- Bạn có thể di chuyển đối tượng hình học được chọn sang trái, phải, lên, xuống 1 pixel bằng cách nhấn phím Ctrl + phím mũi tên (↓, ↑, ←, →) tương ứng. Ví dụ Ctrl + ↓ để di chuyển hình xuống dưới 1 pixel.

- Bạn có thể dùng tổ hợp phím Shift + phím muỗi tên để tiến hành phóng lớn hoặc thu nhỏ ảnh 1 pixel.

- Với mỗi đối tượng hình học, bạn có thể thay đổi độ dày của đường viền, màu của đường viền, màu nền của đối tượng thông qua ba nút nhấn dropdown trên thanh công cụ.

- Các đối tượng hình học được tạo sẽ có thuộc tính mặc định của chương trình. Nhưng bạn có thể vào Options, đánh dấu check vào Create next shape with current formatting. Tùy chọn này sẽ làm cho đối tượng hình học được tạo mới có thuộc tính tượng tự như bạn đã thiết lập cho đối tượng hình học cuối cùng trước khi tạo mới đối tượng này.



Dưới đây, bạn sẽ thấy một hình ảnh ví dụ mẫu cho tất cả các control cũng như cửa sổ trong AutoIt. Chúc bạn sử dụng tốt công cụ Koda FormDesigner trong việc thiết kế giao diện của mình.

Xử lý sự kiện trong AutoIt

I. Tổng quan

Một giao diện GUI có nhiều cửa sổ và mỗi cửa sổ lại có một hoặc nhiều control. Tất nhiên những cửa sổ mà bạn tạo ra là nhằm mục đích thực hiện một tác vụ nào đó chẳng hạn như nhập username và password, xử lý tập tin, xử lý ảnh,… Tất cả những tác vụ đó muốn thực hiện được thì bạn phải dùng code của mình để xử lý, tuy nhiên để xử lý được thì bạn phải biết người dùng đã tiến hành thao tác gì trên bàn phím, trên cửa sổ hoặc sự kiện gì xảy ra trên cửa sổ như cửa sổ được đóng, cửa sổ thu nhỏ trên thanh taskbar, lỗi xảy ra trong khi chạy chương trình,…

Ví dụ : người dùng muốn truy cập vào cơ sở dữ liệu chứa thông tin sinh viên thì trước hết phải nhập tên người dùng và mật khẩu hợp lệ để đăng nhập vào. Hệ quản trị cơ sở dữ liệu muốn biết người dùng đã tiến hành hoàn thành thao tác nhập thông tin tài khoản và mật khẩu chưa thì phải thông qua một sự kiện nào đó, chẳng hạn như người dùng nhấn nút OK hoặc nhấn Enter ngay tại trường gõ nội dung mật khẩu.

Bạn hãy suy nghĩ sự kiện như bạn đang ở trong một ngôi nhà và chờ người đưa thư đến, bạn ngồi ở đó và chờ một người đưa thư bỏ thư vào trong hộp thư trước nhà của mình và sau đó bạn đọc nội dung trong thư và quyết định những gì cần làm với chúng. Đây có thể là một hình ảnh chính xác để minh họa cơ chế xử lý sự kiện của AutoIt. Giống như trong ví dụ trên, hệ quản trị cơ sở dữ liệu phải chờ đợi cho người dùng hoàn thành nhập thông tin tài khoản và mật khẩu thông qua việc người dùng nhấn Enter hoặc bấm OK thì hệ quản trị cơ sở dữ liệu sẽ xem xét mật khẩu đó đã đúng hay chưa rồi sau đó mới cho người dùng tiến hành đăng nhập hay không.

Tất cả những sự kiện xảy ra trong AutoIt sẽ được AutoIt cung cấp cho chúng ta biết được các sự kiện xảy ra và cách xử lý những sự kiện đó. AutoIt hỗ trợ cho chúng ta hai phương pháp để xử lý sự kiện đó là dùng chế độ OnEvent và sử dụng vòng lặp thông điệp. Để hiểu thêm về cơ chế xử lý sự kiện trong AutoIt, bạn có thể xem lại hàm GUIGetMsg, GUISetOnEvent, GUICtrlSetOnEvent, và một số hàm xử lý cửa sổ GUI khác.

II. Xử lý sự kiện bằng cách dùng vòng lặp thông điệp

Trong chế độ vòng lặp thông điệp, sự kiện mà bạn muốn xử lý sẽ được đưa vào một vòng lặp. Trong thân vòng lặp này bạn sẽ phải sử dụng hàm GUIGetMsg() để tiếp nhận những sự kiện nào đó đã xảy ra trên GUI của mình và tương ứng với mỗi sự kiện đó, bạn sẽ phải cung cấp các đoạn code để tiến hành xử lý sự kiện xảy ra. Chế độ vòng lặp thông điệp chính là chế độ xử lý sự kiện mặc định trong AutoIt. Ngoài ra còn có một chế độ khác chính là chế độ OnEvent sẽ trình bày trong phần III.

Dạng chuẩn cơ bản của vòng lặp thông điệp

```
While 1
  $msg = GUIGetMsg()
  ...; đoạn mã xử lý sự kiện sẽ viết ở đây
  ...
WEnd
```

Bạn nhìn vào dạng của chế độ vòng lặp thông điệp trên đây, bạn có thể thấy rằng sẽ gọi rất nhiều lần hàm GUIGetMsg() trong vòng một giây để tiếp nhận sự kiện đã xảy ra. Trong trường hợp bạn không cung cấp các hàm xử lý sự kiện cũng như chỉ thị ExitLoop để thoát khỏi vòng lặp While thì có thể cửa sổ GUI của bạn bị bất động và không đáp ứng lệnh từ người dùng.

Khi bạn chạy vòng lặp như trên để xử lý sự kiện thì rất có thể sẽ chiếm dụng tài nguyên CPU của bạn (bạn có thể xem trong Task Manager để hiểu rõ hơn). Nhưng hàm GUIGetMsg sẽ tự động giảm tài nguyên chiếm dụng cho CPU khi không có sự kiện nào xảy ra. Chính vì thế bạn không nên đặt bất kì một lệnh Sleep hoặc một lệnh tạm dừng nào đó vì có thể trong khoảng thời gian tạm dừng đó nếu có sự kiện gì xảy ra thì đoạn mã lệnh tương ứng xử lý sự kiện đó sẽ không xử lý được.

Hàm GUIGetMsg trong vòng lặp sẽ trả về sự kiện gì đã xảy ra trên cửa sổ. Nó có thể là

- No Event

Khi không có một sự kiện nào đó xảy ra thì hàm GUIGetMsg sẽ trả về 0.

- Control Event

Khi một control được nhấn hoặc thay đổi thì sự kiện tương ứng với nó sẽ được gửi đi. Hàm GUIGetMsg trong trường hợp này sẽ trả về số định danh ID của control tương ứng được trả về trong hàm tạo control GUICtrlCreate…

- System Event

Là các sự kiện hệ thống chẳng hạn như một cửa sổ nào đó đóng. Bạn có thể xem thêm dan sách các sự kiện hệ thống dưới đây. Tất cả những sự kiện này được định nghĩa trong thư viện GUIConstantsEx.au3

```
$GUI_EVENT_CLOSE
$GUI_EVENT_MINIMIZE
$GUI_EVENT_RESTORE
$GUI_EVENT_MAXIMIZE
$GUI_EVENT_PRIMARYDOWN
$GUI_EVENT_PRIMARYUP
$GUI_EVENT_SECONDARYDOWN
$GUI_EVENT_SECONDARYUP
$GUI_EVENT_MOUSEMOVE
$GUI_EVENT_RESIZED
$GUI_EVENT_DROPPED
```

Bạn có thể xem lại hàm GUIGetMsg() để biết rõ thêm các sự kiện này.

Ví dụ xử lý sự kiện dùng vòng lặp thông điệp

Bây giờ trong cửa sổ SciTE bạn hãy tạo một chương trình đơn giản như sau

```
#include <GUIConstantsEx.au3>

GUICreate("Trac nghiem tinh yeu", 200, 100)

GUICtrlCreateLabel("Em co yeu anh khong ?", 30, 10)

$btnok = GUICtrlCreateButton("Co", 40, 50, 60)

$btnno = GUICtrlCreateButton("Khong", 100, 50, 60)
```

```
GUISetState(@SW_SHOW)
Sleep(1000)
```

Bây giờ chúng ta sẽ tiến hành thêm một đoạn code xử lý sự kiện dùng vòng lặp thông điệp để xử lý một số sự kiện đơn giản như sau.

```
include <GUIConstantsEx.au3>
GUICreate("Trac nghiem tinh yeu", 200, 100)
GUICtrlCreateLabel("Em co yeu anh khong ?", 30, 10)
$btnok = GUICtrlCreateButton("Co", 40, 50, 60)
$btnno = GUICtrlCreateButton("Khong", 100, 50, 60)
GUISetState(@SW_SHOW)
While 1
  $msg = GUIGetMsg()
        Select
                Case $msg = $btnok
                  MsgBox(0, "Giai dap", "Cam on vi em da chap nhan anh !")
                Case $msg = $btnno
                        MsgBox(0, "Giai dap", "Chuc em hanh phuc !")
                Case $msg = $GUI_EVENT_CLOSE
                  MsgBox(0, "Tro choi ket thuc", "Ban da nhan nut Close de thoat chuong
trinh")
                ExitLoop
        EndSelect
WEnd
Sleep(1000)
```

Một ví dụ thật đơn giản cũng như dễ hiểu để minh hoạt cho chế độ xử lý sự kiện dùng vòng lặp thông điệp. Khi bạn thành thạo cái cơ bản này, bạn có thể tự tìm hiểu để nâng cao khả năng xử lý các sự kiện khác.

Cũng xin lưu ý với bạn rằng, mỗi control bạn tạo trên cửa sổ đều có một số định danh ID duy nhất được tạo bởi các hàm tạo control, cho dù bạn tạo nhiều control khác nhau trên cửa sổ khác nhau. Bạn có thể dễ dàng thiết lập xử lý sự kiện tương ứng với control phát ra sự kiện thông qua hàm GUIGetMsg(). Tuy nhiên, nếu bạn tạo nhiều cửa sổ khác nhau thì khi xảy ra sự kiện trên cửa sổ thì bạn khó có thể đoán biết được cửa sổ nào xảy ra sự kiện đó. Trong ví dụ trên bạn thấy rằng ta chỉ tạo một cửa sổ nên trong vòng lặp ta đã thiết lập lệnh Select Case tương ứng với sự kiện $GUI_EVENT_CLOSE là hiển thị một hộp thoại thông báo và kết thúc vòng lặp. Tuy nhiên nếu bạn tạo ra hai, ba hoặc nhiều cửa sổ hơn thì sao. Làm sao chúng ta biết được cửa sổ nào xảy ra sự kiện. Điều này có thể giải quyết bằng cách thiết lập hàm GUIGetMsg với tham số là 1 (GUIGetMsg(1)). Khi gọi hàm này với tham số là 1, giá trị trả về của hàm này sẽ là một mảng có 5 phần tử. Trong đó sự kiện nào xảy ra tương ứng với control hoặc cửa sổ nào đó sẽ được lưu tại vị trí đầu tiên. Phần tử thứ hai sẽ chứa biến "handle" chỉ cửa sổ đã phát ra sự kiện đó. Ba phần tử còn lại, bạn hãy xem thêm hàm GUIGetMsg trong các bài viết kì trước.

Bây giờ để minh họa chúng ta sẽ minh họa lại ví dụ trên thông qua đoạn code như sau

```
#include <GUIConstantsEx.au3>
```

```
$main = GUICreate("Trac nghiem tinh yeu", 200, 100)
GUICtrlCreateLabel("Em co yeu anh khong ?", 30, 10)
$btnok = GUICtrlCreateButton("Co", 40, 50, 60)
$btnno = GUICtrlCreateButton("Khong", 100, 50, 60)
$main2 = GUICreate("Cua so tam", 200, 100, 150, 200)
GUISetState(@SW_SHOW, $main)
GUISetState(@SW_SHOW, $main2)
While 1
  $msg = GUIGetMsg(1)
        Select
                Case $msg[0] = $btnok
                  MsgBox(0, "Giai dap", "Cam on vi em da chap nhan anh !")
                Case $msg[0] = $btnno
                        MsgBox(0, "Giai dap", "Chuc em hanh phuc !")
                Case $msg[0] = $GUI_EVENT_CLOSE And $msg[1] = $main
                  MsgBox(0, "Tro choi ket thuc", "Ban da nhan nut Close de thoat chuong
trinh")
                ExitLoop
                Case $msg[0] = $GUI_EVENT_MINIMIZE And $msg[1]= $main2
                  MsgBox(0, "Trac nghiem tinh yeu", "Cua so thu hai da thu nho tren he
thong")
                ExitLoop
        EndSelect
WEnd
Sleep(1000)
```

Trong ví dụ trên, tôi tạo hai cửa sổ $main và $main2. Sau đó trong vòng lặp thông điệp tôi gán cho sự kiện cửa sổ chương trình chính ($main) đóng thì xuất hiện thông báo "Ban da nhan nut Close de thoat chuong trinh". Trong khi đó cửa sổ tạm ($main2) tôi gán sự kiện khi nào cửa sổ này thu nhỏ trên thanh taskbar thì sẽ có một thông báo "*Cua so thu hai da thu nho tren he thong".*

Nói chung việc dùng vòng lặp thông điệp này có một hạn chế là khi chương trình chạy tới vòng lặp này thì đoạn mã lệnh sau vòng lặp thông điệp sẽ không thể chạy được. Việc dùng vòng lặp thông điệp sẽ có hiệu quả hơn trong trường hợp bạn thiết lập vòng lặp xử lý thông điệp cho một thời điểm nào đó của chương trình đang chạy khi chắc chắn sẽ có một sự kiện gì đó xảy ra trên cửa sổ GUI của bạn.

III. Xử lý sự kiện dùng chế độ OnEvent

Không giống như vòng lặp thông điệp, chế độ OnEvent sẽ gán một hàm tương ứng do người dùng tạo ra với sự kiện xảy ra tương ứng trên control và GUI. Đến khi sự kiện xảy ra trên GUI hoặc control thì đoạn chương trình chính sẽ tạm dừng và gọi hàm tương ứng để xử lý sự kiện đó. Ví dụ khi bạn nhấn vào một control Button thì đoạn chương trình chính sẽ dừng lại và gọi hàm trước đó mà người dùng đã thiết lập cho control Button này. Sau khi hàm xử lý sự kiện kết thúc thì đoạn mã lệnh của chương trình chính của bạn sẽ chạy tiếp. Chế độ này tương tự xử lý sự kiện trong Visual Basic.NET hoặc trong C#. Để xử lý sự kiện trong chế độ OnEvent mời bạn xem lại hai hàm GUISetOnEvent (xử

lý sự kiện trên GUI) và GUICtrlSetOnEvent (xử lý sự kiện cho control). Bên trong các hàm xử lý sự kiện, bạn có thể sử dụng hai macro @GUI_CTRLID (ID của control gửi thông điệp hoặc ID của sự kiện hệ thống) và @GUI_WINHANDLE (biến "handle" của cửa sổ xảy ra thông điệp) để hỗ trợ xử lý các sự kiện.

Chú ý : Để xử lý sự kiện trong chế độ OnEvent, bạn phải thêm chỉ thị Opt("GUIOnEventMode", 1) ngay tại phần đầu chương trình.

Chúng ta sẽ minh họa cho chế độ này bằng ví dụ sau

```
#include <GUIConstantsEx.au3>
Opt("GUIOnEventMode", 1)
$main = GUICreate("Trac nghiem tinh yeu", 200, 100)
GUICtrlCreateLabel("Em co yeu anh khong ?", 30, 10)
$btnok = GUICtrlCreateButton("Co", 40, 50, 60)
GUICtrlSetOnEvent($btnok, "OKPress")
$btnno = GUICtrlCreateButton("Khong", 100, 50, 60)
GUICtrlSetOnEvent($btnno, "NoPress")
$main2 = GUICreate("Cua so tam", 200, 100, 150, 200)
GUISetOnEvent($GUI_EVENT_CLOSE, "CloseWin", $main)
GUISetOnEvent($GUI_EVENT_MINIMIZE, "MiniWin", $main2)
GUISetState(@SW_SHOW, $main)
GUISetState(@SW_SHOW, $main2)
Sleep(12000)
Func OKPress()
        MsgBox(0, "Giai dap", "Cam on vi em da chap nhan anh !")
EndFunc
Func NoPress()
        MsgBox(0, "Giai dap", "Chuc em hanh phuc !")
EndFunc
Func CloseWin()
                MsgBox(0, "Tro choi ket thuc", "Ban da nhan nut Close de thoat chuong trinh")

EndFunc
Func MiniWin()
                MsgBox(0, "Trac nghiem tinh yeu", "Cua so thu hai da thu nho tren he thong")
EndFunc
```

Đoạn chương trình này có ý nghĩa tương tự như chương trình ví dụ minh họa trong vòng lặp thông điệp nhưng được viết lại dưới dạng xử lý sự kiện theo chế độ OnEvent.

Ưu điểm của dùng chế độ OnEvent là đoạn code sau đoạn thiết lập hàm xử lý sự kiện như GUISetOnEvent và GUICtrlSetOnEvent vẫn chạy, các hàm đó chỉ gán hàm tương ứng với sự kiện xảy ra trên control hay GUI chứ không giống như vòng lặp thông điệp là

đoạn code sau vòng lặp thông điệp bị tạm dừng cho đến khi thoát khỏi vòng lặp.

Hàm giả lập thao tác chuột máy tính

1. MouseClick("button", [x], [y], [click], [speed])

- Công dụng : giả lập một thao tác nhấn chuột.

- button là tham số chỉ định nút trên chuột sẽ được nhấn. Thông thường sẽ có những tham số như sau : left hoặc primary hoặc main (nút trái chuột sẽ được nhấn), right hoặc secondary hoặc menu (nút phải chuột sẽ được nhấn), middle (nút ở giữa con chuột, nó nằm giữa nút trái và nút phải chuột. Chú ý là thao tác nhấn chứ không phải là cuộn lên cuộn xuống).

- x, y là tọa độ trên màn hình mà con trỏ chuột sẽ di chuyển tới trước khi nhấn. Nếu tham số x, y không được cho, vị trí hiện tại của con trỏ chuột sẽ được sử dụng.

- click : số lần thực hiện thao tác nhấn chuột. Mặc định là 1.

- speed : tốc độ di chuyển chuột, nó có giá trị từ 1 (nhanh nhất) đến 100 (chậm nhất). Nếu bạn gán tham số này là 0 thì con chuột sẽ di chuyển nhanh tức thì. Giá trị mặc định của tham số này là 10.

- Nếu người dùng cung cấp tham số button là "" thì chuột trái sẽ được nhấn.

- Nếu người dùng tiến hành tráo đổi vai trò của các nút trái hoặc nút phải chuột trong Control Panel (áp dụng cho người dùng chuột bằng tay trái) thì sẽ có một chút thay đổi khi thực hiện hàm này. Cụ thể bạn xem trong bảng sau :

Button	Chuột bình thường, không tráo đổi vai trò giữa nút trái và nút phải chuột	Chuột sau khi tráo đổi vai trò giữa nút trái và nút phải chuột trong Control Panel
""	Nút trái chuột sẽ được nhấn	Nút trái chuột sẽ được nhấn
"left"	Nút trái chuột sẽ được nhấn	Nút trái chuột sẽ được nhấn
"middle"	Nút giữa chuột sẽ được nhấn	Nút giữa chuột sẽ được nhấn
"right"	Nút phải chuột sẽ được nhấn	Nút phải chuột sẽ được nhấn
"primary"	Nút trái chuột sẽ được nhấn	Nút phải chuột sẽ được nhấn
"main"	Nút trái chuột sẽ được nhấn	Nút phải chuột sẽ được nhấn
"secondary"	Nút phải chuột sẽ được nhấn	Nút trái chuột sẽ được nhấn
"menu"	Nút phải chuột sẽ được nhấn	Nút trái chuột sẽ được nhấn

Ví dụ : *MouseClick("left", 0, 500, 1)*

2. MouseClickDrag ("button", x1, y1, x2, y2, [speed])

- Công dụng : thực hiện thao tác nhấn chuột và giữ chuột (drag) từ vị trí này đến vị trí khác.

- x1, y1 : là tọa độ trên màn hình mà chuột tiến hành bắt đầu thao tác giữ chuột. x2, y2

: là tọa độ trên màn hình mà chuột sẽ di chuyển tới cuối cùng trong quá trình giữ chuột.

- Tham số speed, button, cũng như những tác động của hàm khi thay đổi vai trò của nút trái chuột và nút phải chuột trong ControlPanel tương tự hàm MouseClick.

Ví dụ : *MouseClickDrag("right", 0, 200, 600, 700)*

3. MouseDown ("button")

- Công dụng : thực hiện thao tác nhấn giữ chuột (nhấn chìm một nút nào đó của chuột) tại vị trí con trỏ chuột hiện hành.

- Tham số button tương tự như hàm MouseClick.

Ví dụ : *MouseDown("left")*

Sleep(2000)

4. MouseUp("button")

- Công dụng : thực hiện thao tác nhả chuột ra tại vị trí con trỏ chuột hiện hành. Hàm này nên đi kèm với MouseDown để thực hiện thao tác nhấn và nhả chuột.

- Tham số button tương tự như hàm MouseClick.

Ví dụ : *MouseDown("right")*

Sleep(2000)

MouseUp("right")

5. MouseMove (x, y, [speed])

- Công dụng : thực hiện thao tác di chuyển chuột bắt đầu từ vị trí con trỏ chuột hiện tại.

- x, y là tọa độ đích trên màn hình mà con trỏ chuột sẽ di chuyển tới.

- Tham số speed tương tự như hàm MouseClick.

Ví dụ : *MouseMove(10, 100)*

MouseMove(700, 700, 0)

6. MouseGetPos([dimension])

- Công dụng : lấy thông tin tọa độ X, Y của con trỏ chuột hiện tại trên màn hình.

- dimension là tham số quy định giá trị sẽ trả về. Nếu bạn không cung cấp tham số này, hàm sẽ trả về một mảng có hai phần tử, trong đó phần tử đầu tiền chứa tọa độ X, phần tử thứ hai chứa tọa độ Y. Nếu tham số này là 0, hàm chỉ trả về tọa độ X. Nếu tham số này là 1, hàm chỉ sẽ trả về tọa độ Y.

Ví dụ : *$pos = MouseGetPos()*

MsgBox(0, "Vi tri con tro chuot hien tai", $pos[0] & " - " & $pos[1])

7. MouseWheel("direction", [click])

- Công dụng : thực hiện thao tác cuộn chuột lên hoặc cuộn chuột xuống bằng nút giữa của con chuột.

- direction : là tham số quy định thao tác cuộn lên hoặc cuộn xuống. Nếu tham số này là up sẽ tiến hành cuộn lên. Nếu tham số này down sẽ tiến hành cuộn xuống. click là số lần thao tác tiến hành cuộn lên hoặc cuộn xuống.

Ví dụ : *MouseWheel("up", 10)*; thực hiện thao tác cuộn lên 10 lần

Từ khóa / câu lệnh trong AutoIt

Bài viết của tác giả Võ Minh Trí – DH7TH2 có cung cấp một số từ khóa / câu lệnh đơn giản trong AutoIt. Nay tôi xin giới thiệu thêm cho bạn một số từ khóa / câu lệnh khác.

1. #comments-start và #comments-end

- Công dụng : xác lập một đoạn nào đó trong mã lệnh đang soạn thảo thành chú thích.

Ví dụ : *#comments-start*

MsgBox(4096, "", "This won't be executed")

MsgBox(4096, "", "Or this")

#comments-end

Đoạn mã trên sẽ làm cho hai câu lệnh MsgBox trở thành chú thích chứ không phải là mã lệnh thực thi nữa.

- Bạn có thể thay thế từ khóa #comments-start thành #cs và #comments-end thành ce cũng có công dụng tạo khối chú thích tương tự.

2. #include

- Công dụng : sử dụng một thư viện nằm trong thư mục Include của AutoIt nhằm mục đích sử dụng một hàm nằm trong một thư viện mà chúng ta đã include vào.

- Cú pháp : #include "[path\]filename" hoặc #include <filename>. Trong đó filename là tên tập tin cần include vào. path là một tham số tùy chọn là đường dẫn đến tập tin cần include. Nhưng nó phải làm một chuỗi chứ không phải là một biến. Nếu bạn sử dụng chỉ thị include dạng #include "[path\]filename", bạn phải cung cấp đường dẫn đầy đủ tập tin cần include vào. Nếu bạn sử dụng chỉ thị include dạng #include <filename> thì tập tin thư viện được nạp sẽ được AutoIt lấy từ thư mục thư viện của AutoIt (thông thường là C:\Program Files\AutoIt3\Include. Thư viện của AutoIt có nhiều hàm được định nghĩa sẵn để cho bạn dùng.

Ví dụ : Để nạp thư viện GUIConstantsEx.au3 bạn có thể dùng chỉ thị *#include <GUIConstantsEx.au3>* ngay tại phần đầu chương trình hoặc nếu thư viện GUIConstantsEx.au3 nằm trong phân vùng E bạn có thể dùng *#include "E:\GUIConstants.au3"*

- Trong AutoIt, một script khác (*.au3) có thể được include vào chương trình đang soạn thảo bằng chỉ thị #include.

3. #include-once

- Công dụng : nếu bạn include vào một tập tin thư viện mà tập tin thư viện đó có chứa hàm trùng tên thì sẽ báo lỗi "Duplicate function". Do đó để tránh lỗi trên, bạn phải thêm chỉ thị #include-once để ngăn cản lỗi nạp chồng hàm.

- Để sử dụng, bạn chỉ cần thêm chr thị #include-once ngay tại phần đầu chương trình.

4. #NoTrayIcon

- Công dụng : không cho hiển thị biểu tượng của AutoIt trên khay hệ thống khi chương trình đang chạy.

- Bạn cũng có thể sử dụng mã Opt("TrayIconHide", 1) trong chương trình để gỡ bỏ biểu tượng của AutoIt trên khay hệ thống, tuy nhiên biểu tượng của AutoIt sẽ vẫn hiển thị trong vòng một giây khi chương trình bắt đầu. Để sử dụng từ khóa này, bạn hãy đặt từ khóa này ngay tại phần đầu chương trình của mình.

- Bạn vẫn có thể tiến hành cho hiện icon AutoIt trên khay hệ thống bằng cách sử dụng mã Opt("TrayIconHide", 0).

Ví dụ : *#NoTrayIcon*

MsgBox(4096,"Nhan OK","Hien icon tren khay he thong trong vong 5 giay...")

Opt("TrayIconHide", 0) ; ẩn icon của AutoIt

Sleep(5000)

5. #OnAutoItStartRegister "function"

- Công dụng : gọi một hàm do người dùng định nghĩa khi chương trình mã lệnh AutoIt của bạn bắt đầu thực thi. function là tên hàm mà bạn cần gọi.

Ví dụ : *#OnAutoItStartRegister "MyTestFunc"*

```
MsgBox(0, "MyTest", "Hello World")
Func MyTestFunc()
        MsgBox(0, "AutoIt Start", "AutoIt Start")
EndFunc
```

6. #RequireAdmin

- Công dụng : thiết lập đoạn script hiện tại yêu cầu tài khoản người dùng phải có quyền quản trị mới được chạy.

- Từ khóa này của chương trình nhằm mục đích làm cho ngôn ngữ AutoIt chạy đúng trong hệ điều hành Windows Vista (UAC). Tuy nhiên nó cũng làm việc tốt với Windows 2000 và Windows XP.

Ví dụ : *#RequireAdmin*

```
MsgBox(4096,"Info","Now running with admin rights")
```

7. ContinueCase

- Công dụng : kết thúc case hiện tại và thực hiện câu lệnh của case kế tiếp trong cấu trúc Select hoặc Switch

- Thông thường trong cấu trúc Select hoặc Switch, thì biểu thức trong case thứ nhất không đúng thì sẽ kiểm tra mệnh đề case thứ hai, nếu case thứ hai không đúng thì thực thi kiểm tra mệnh đề của case thứ ba và cho đến hết case. Nếu một trong các mệnh đề của case nào đó đúng thì sẽ thực thi câu lệnh tương ứng với case đó và kết thúc thực thi cấu trúc Select hoặc Switch. Việc thêm câu lệnh ContinueCase sẽ làm cho case hiện tại kết thúc và chuyển sang thực thi câu lệnh của case thứ hai.

- Việc thực thi câu lệnh ContinueCase bên ngoài cấu trúc Select hoặc Switch sẽ gây ra lỗi.

Ví dụ :

```
$msg = ""
$szName = InputBox(Default, "Please enter a word.", "", " M", _
Default, Default, Default, Default, 10)
Switch @error
Case 2
   $msg = "Timeout "
   ContinueCase
Case 1; Continuing previous case
   $msg &= "Cancellation"
Case 0
   Switch $szName
   Case "a", "e", "i", "o", "u"
      $msg = "Vowel"
   Case "QP"
```

```
        $msg = "Mathematics"
    Case "Q" to "QZ"
        $msg = "Science"
    Case Else
        $msg = "Others"
    EndSwitch
Case Else
    $msg = "Something went horribly wrong."
EndSwitch
MsgBox(0, Default, $msg)
```

8. ContinueLoop

- Công dụng : chạy tiếp vòng lặp tiếp theo trong các vòng lặp While, Do, For.

- ContinueLoop sẽ bỏ qua vòng lặp hiện tại và tiếp tục thực thi các biểu thức kiểm tra trong các vòng lặp (đó là biểu thức trong các lệnh While, Until hoặc Next). Mặc dù đoạn chương trình dùng câu lệnh ContinueLoop có thể viết lại bằng cấu trúc If. Tuy nhiên việc dùng ContinueLoop có thể làm cho chương trình của bạn dễ hiểu hơn. Khi sử dụng câu lệnh ContinueLoop bạn phải chú ý sử dụng cho đúng nếu không sẽ tạo nên các vòng lặp vô tận, đặc biệt là trong vòng lặp While/Do.

Ví dụ :
```
For $i = 1 to 10
    If $i = 7 Then ContinueLoop
        MsgBox(0, "Gia tri cua $i la:", $i)
    Next
```

Đoạn chương trình trên sẽ in giá trị của biến $i từ 1 đến 10 nhưng không in giá trị 7.

9. Dim

- Công dụng : khai báo biến, hằng, hoặc tạo mảng.

- Cú pháp khai báo như sau :

Dim [Const] $variable [= initializer] hoặc để khai báo mảng ta dùng

Dim [Const] $array[subscript 1]...[subscript n] [= initializer]

- Tham số const là tham số tùy chọn, nếu có thì sẽ tạo ra một biến hằng và không thể thay đổi giá trị của nó. $variable là tên biến cần khai báo. initializer là giá trị khởi tạo cho một biến. subscript là số phần tử tương với cho một chiều nào đó của mảng.

- Các từ khóa như Dim, Local và Global thực hiện các chức năng tương tự như : khởi tạo một biến trước khi bạn dùng nó (tương tự như VBScript) và tạo một mảng.

- Trong AutoIt, bạn có thể tạo ra một biến đơn giản bằng cách gán biến đó với một giá trị, chẳng hạn như &myvar = 0 nhưng nhiều người muốn khai báo rõ ràng hơn. Nếu bạn cung cấp đoạn mã Opt("MustDeclareVars", 1) ngay tại phần đầu chương trình thì các biến phải được khai báo trước khi sử dụng. Bạn có thể khai báo biến và khởi tạo giá trị của biến trên một dòng như ví dụ sau :

```
Dim $a = 2, $b = 10, $c = 20
```

- Để khởi tạo giá trị cho một mảng nhiều chiều, bạn hãy đặt giá trị của mỗi phần tử bên trong dấu ngoặc vuông, cách nhau bằng dấu phẩy. Đối với mảng nhiều chiều, bạn có thể cung cấp số giá trị thấp hơn số phần từ mà bạn đã khai báo, nhưng không được nhiều hơn.

Ví dụ : *Dim $arrayM[2][5] = [[1,2,3,4,5], ["a","b"]]*

- AutoIt có khả năng cho phép bạn sao chép một mảng này sang mảng khác, chẳng hạn như $mycopy = $myarray. Trong câu lệnh này sẽ sao chép nội dung của mảng $myarray sang $mycopy và biến $mycopy sẽ có cùng số chiều với mảng $myarray. Để xóa nội dung của một mảng, bạn có thể chỉ cung cấp một lệnh gán đơn giản như sau $array = 0. Hàm này sẽ hủy nội dung của mảng và chuyển nó về biến thông thường có giá trị là 0.

- Nếu bạn khai báo một biến trong một hàm cùng tên với một tham số của hàm đó bằng cách dùng chỉ thị Local thì sẽ có một lỗi xảy ra. Bạn có thể dùng chỉ thị Global để gán một biến trong hàm làm một biến toàn cục, tuy nhiên nếu biến local (hay tham số) của hàm trùng tên với biến Global nào đó thì chỉ có biến local được sử dụng mà thôi. Bạn nên đặt tên các biến local và global có tên khác nhau.

10. Enum

- Công dụng : sử dụng các biến để lưu trữ một tập hợp, chẳng hạn như các ngày trong tuần và các tháng trong năm.

- Cú pháp [scope] Enum [Step <stepval>] <constantlist>. Trong đó scope là dạng biến cho tập hợp Enum, nó có thể là Dim, Local hoặc Global. Nếu bạn không cung cấp, Dim sẽ mặc định được sử dụng. stepval là giá trị nhảy của biến trong tập hợp, mặc định là 1. Bạn có thể sử dụng các phương thức để nhảy như *n, +n, -n với n là một số nguyên. constantlist là danh sách các biến chứa hằng để liệt kê cách nhau bằng dấu phẩy.

- Theo mặc định, biến đầu tiên trong bộ liệt kê sẽ có giá trị là 0, các biến liệt kê còn lại sẽ có giá trị bằng giá trị của biến trước đó cộng với 1. Khi bạn cung cấp tham số tăng giá trị trong stepval thì biến hằng đầu tiên sẽ gán giá trị là 1, các phần tử còn lại sẽ có giá trị bằng giá trị của phần tử trước đó tăng theo giá trị bước nhảy stepval mà bạn đã quy định. Biến liệt kê trong Enum có thể có giá trị được gán bởi một câu lệnh hợp lệ từ người dùng.

Ví dụ : *Global Enum $E1VAR1, $E1VAR2, $E1VAR3*

MsgBox(4096, "", "$E1VAR1 = " & $E1VAR1); trả về 0

MsgBox(4096, "", "$E1VAR2 = " & $E1VAR2); trả về 1

MsgBox(4096, "", "$E1VAR3 = " & $E1VAR3); trả về 2

Global Enum $E2VAR1 = 10, $E2VAR2, $E2VAR3 = 15

MsgBox(4096, "", "$E2VAR1 = " & $E2VAR1); trả về 10

MsgBox(4096, "", "$E2VAR2 = " & $E2VAR2); trả về 11

MsgBox(4096, "", "$E2VAR3 = " & $E2VAR3); trả về 15

*Global Enum Step *2 $E3VAR1, $E3VAR2, $E3VAR3*

MsgBox(4096, "", "$E3VAR1 = " & $E3VAR1); trả về 1

MsgBox(4096, "", "$E3VAR2 = " & $E3VAR2); trả về 2

MsgBox(4096, "", "$E3VAR3 = " & $E3VAR3); trả về 4

11. ReDim

- Công dụng : xác lập lại kích thước của một mảng

- Cú pháp : ReDim $array[subscript 1]...[subscript n]. Trong đó array là tên biến mảng cần thay đổi kích thước. subscript 1 đến subscript n là số phần tử mới của chiều tương ứng trong mảng đó.

- Từ khóa ReDim tương tự như Dim, ngoại trừ việc ReDim giữ lại các giá trị trong mảng thay vì loại bỏ các giá trị trong khi thay đổi kích thước của một mảng. Số chiều của mảng vẫn như cũ đồng thời giữ lại phạm vi của biến mảng (Global hoặc Local) mà nó đã

có trước khi thay đổi kích thước.

Ví dụ :
```
Dim $I, $K, $T, $MSG
Dim $X[4][6], $Y[4][6]
For $I = 0 To 3
  For $K = 0 To 5
    $T = Int(Random(20) + 1)  ; trả về số ngẫu nhiên từ 1 đến 20
    $X[$I][$K] = $T
    $Y[$I][$K] = $T
  Next
Next
ReDim $X[3][8]
Dim $Y[3][8]
$MSG = ""
; hiển thị các giá trị của phần tử mảng X
For $I = 0 To UBound($X, 1) - 1
  For $K = 0 To UBound($X, 2) - 1
    If $K > 0 Then $MSG = $MSG & ", "
    $MSG = $MSG & $X[$I][$K]
  Next
  $MSG = $MSG & @CR
Next
MsgBox(0, "ReDim Demo", $MSG)
$MSG = ""
; hiển thị các giá trị của phần tử mảng X
For $I = 0 To UBound($Y, 1) - 1
  For $K = 0 To UBound($Y, 2) - 1
    If $K > 0 Then $MSG = $MSG & ", "
    $MSG = $MSG & $Y[$I][$K]
  Next
  $MSG = $MSG & @CR
Next
MsgBox(0, "ReDim Demo", $MSG)
```

Ngay sau khi bạn chạy ví dụ trên, bạn sẽ thấy phần tử của mảng X vẫn giữ lại giá trị sau khi thay đổi kích thước. Riêng mảng Y do khai báo với từ khóa Dim nên các giá trị của mảng đã bị xóa sạch hết sau khi thay đổi kích thước và do đó bạn sẽ thấy khi hiển thị phần tử của mảng Y sẽ không có gì hết.

12. ExitLoop [level]

- Công dụng : thoát khỏi vòng lặp While/Do/For.
- level là số bậc của vòng lặp cần thoát. Tham số này thường được áp dụng cho các vòng

lặp lồng nhau. Mặc định là 1, tức là vòng lặp chứa chỉ thị ExitLoop sẽ thoát.

Ví dụ : *$sum = 0*

```
While 1 = 1
    While 1 ;dùng vòng lặp vô lận cho đến khi chỉ thị ExitLoop được gọi
        $ans = InputBox("Running total=" & $sum, _
            "  Enter a positive number.  (A negative number exits)")
        If $ans < 0 Then ExitLoop 2
        $sum = $sum + $ans
    WEnd
    MsgBox(0, "ExitLoop", "Da Thoat vong lap ""while 1""")
WEnd
MsgBox(0,"The sum was", $sum)
```

Đoạn chương trình trên sẽ cho người dùng nhập các con số vào hộp InputBox sau đó tính tổng các số vừa nhập vào. Chương trình bắt đầu hiển thị kết quả tổng sau khi người dùng nhập số âm. Trong chỉ thị ExitLoop 2 bạn thấy trong ví dụ, chỉ thị đó sẽ làm cho vòng lặp "While 1 =1" thoát.

Những câu hỏi thường gặp khi lập trình AutoIt

Phần này trình bày cho bạn một số câu hỏi thường gặp khi sử dụng ngôn ngữ lập trình AutoIt trên diễn đàn của những người sử dụng AutoIt trên trang http://www.autoitscript.com/forum/. Nếu bạn không tìm thấy những câu hỏi mình thắc mắc ở đây, bạn có thể vào trang web ở trên để tham gia thảo luận với các thành viên khác của diễn đàn.

1. Tại sao tôi không thể chạy những đoạn mã AutoIt v2.64 trên phiên bản v3 ?

Phiên bản v3 của AutoIt có một số khác biệt về cấu trúc ngôn ngữ so với phiển bản v2.64. Các phiên bản trước đó của AutoIt thường được dùng để viết các đoạn mã lệnh đơn giản nhằm mục đích cài đặt phần mềm. Tuy nhiên theo thời gian, người dùng AutoIt đã dùng mã lệnh của ngôn ngữ này để lập trình cho các tác vụ tự động hóa ngày càng phức tạp. Cú pháp và cấu trúc của ngôn ngữ trong phiên bản cũ của AutoIt có thể thực hiện những tác vụ đó tuy nhiên nó gây ra một số khó khăn cho những người lập trình và khá rườm rà. Để cho ngôn ngữ AutoIt có thể thực hiện các tác vụ tự động hóa nói chung và phù hợp hơn thì phiên bản mới của AutoIt v3 đã ra đời. Và bây giờ nếu bạn có một nền tảng lập trình trên các ngôn ngữ khác, bạn có thể sử dụng AutoIt v3 này một cách dễ dàng.

2. Phiên bản v3 của AutoIt có gây khó khăn cho những người lập trình hơn so với các phiên bản trước không ?

Hầu hết trong mọi trường hợp, phiên bản v3 của AutoIt sử dụng dễ dàng hơn so với các phiên bản trước đó. Nó cũng tương đồng với ngôn ngữ BASIC và như bạn biết đấy BASIC là một ngôn ngữ khá đơn giản. Phần lớn các đoạn mã lệnh AutoIt phiên bản cũ chủ yếu tập trung thao tác nhấn "Next" trên các hộp thoại trong việc cài đặt phần mềm. Hầu hết các đoạn mã lệnh thực hiện thao tác đó có thể dễ dàng chuyển đổi sang v3 chỉ đơn giản bằng cách thêm một dấu ngoặc. Dưới đây là một ví dụ cho bạn thấy sự thay đổi như vậy trong phiên bản v2 và v3. Đoạn mã sau đây minh họa việc tự động cài đặt phần mềm thông qua các thao tác nhấn "Next" và kết thúc cài đặt thông qua nhấn nút "Finish" trên hộp thoại cài đặt.

; Đoạn mã lệnh trên phiên bản v2.64

WinWaitActive, Welcome, Welcome to the XSoft installation

Send, !n

WinWaitActive, Choose Destination, Please choose the

Send, !n

WinWaitActive, Ready to install, Click Next to install

Send, !n

WinWaitActive, Installation Complete, Click Finish to exit

Send, !f

WinWaitClose, Installation Complete

; đoạn mã lệnh tương ứng trên phiên bản v3
```
WinWaitActive("Welcome", "Welcome to the XSoft installation")
Send("!n")
WinWaitActive("Choose Destination", "Please choose the")
Send("!n")
WinWaitActive("Ready to install", "Click Next to install")
Send("!n")
WinWaitActive("Installation Complete", "Click Finish to exit")
Send("!f")
WinWaitClose("Installation Complete")
```

Bạn có thể thấy rằng phiên bản v3 giúp cho người dùng thuận tiện trong khi sử dụng hàm chứ không phải lung tung như phiên bản v2.64. Tất cả các chuỗi đều được đóng trong dấu ngoặc kép và bây giờ các tham số trong hàm được đặt trong dấu ngoặc kép và được ngăn cách bởi dấu ",". Hầu hết các trình soạn thảo ngôn ngữ đơn giản đều có hỗ trợ làm nổi bật cú pháp bằng tô sáng (highlight) và vì thế bạn có thể soạn thảo mã đơn giản hơn rất nhiều.

3. Tôi có thể chuyển đổi đoạn mã lệnh AutoIt v2 sang phiên bản v3 bằng cách nào ?

Vấn đề đầu tiên bạn cần hãy suy nghĩ xem bạn có cần phải chuyển đổi đoạn mã của bạn không. V2.64 vẫn có thể tải về trên mạng và còn được hỗ trợ và vì thế bạn không cần phải chuyển đổi các đoạn mã lệnh của bạn, ngoại trừ trường hợp bạn muốn như thế.

Trong tài liệu AutoIt Help có một chương nói về mối liên hệ của dòng lệnh trong phiên bản AutoIt v2 và v3. Bạn có thể xem trong mục Using AutoIt -> Note for Auto It v2 Users trong tài liệu Help của AutoIt.

Các tác giả của AutoIt v3 có viết một tiện ích chuyển đổi đoạn mã phiên bản v2 sang v3. Tiện ích này làm việc khá tốt, bạn có thể tìm tiện ích này trong thư mục "Extras" trong thư mục cài đặt của AutoIt. Tuy nhiên theo tôi thì không có tiện ích chuyển đổi nào trong phiên bản AutoIt mà tôi đang sử dụng, rất có thể nó nằm trong phiên bản v3 cũ hơn. Bạn có thể lên Google Search xem coi có tiện ích chuyển đổi tương tự như vậy không.

4. Chỉ thị lệnh "goto" nằm ở đâu ?

Chỉ thị goto gây ra nhiều lỗi khó kiểm soát trong khi lập trình. Việc sử dụng chỉ thị goto có thể làm mất cấu trúc của chương trình (như lời của With – tác giả của Pascal có nói). Đối với các chương trình nhỏ, bạn có thể dùng chỉ thị goto. Tuy nhiên đối với các chương trình lớn, việc dùng goto sẽ làm cho chương trình thêm phức tạp.

AutoIt có hỗ trợ cho người lập trình dùng một số vòng lặp phổ biến và vì thế chỉ thị goto không còn cần thiết để sử dụng nữa. Bạn có thể sử dụng các chỉ thị mà AutoIt hỗ trợ như While, Do, For, ExitLoop, ContinueLoop. Khi bạn nắm được các vòng lặp này, bạn có thể hiểu được các cấu trúc lặp trong các ngôn ngữ khác chỉ trong vòng một vài phút.

Để minh họa cho việc dùng vòng lặp có thể thay thế chỉ thị goto, bạn có thể xem ví dụ sau :

Trong phiên bản v2.64, tôi sẽ minh họa dùng vòng lặp vô tận như sau

```
:mylabel
...do something...

...and something else...

goto, mylabel
```

Trong phiên bản v3, bạn có thể thay vòng lặp trên bằng cách dùng While tuy nhiên điều kiện vòng lặp luôn luôn đúng.

```
While 1 = 1
  ...do something...
  ...do something else...
Wend
```

5. Những giới hạn trong ngôn ngữ AutoIt V3 ?

Dưới đây sẽ trình bày cho bạn một số giới hạn trong ngôn ngữ AutoIt. Bạn nên nhớ rằng tất cả những cái trình bày sau đây chỉ mang tính lý thuyết và các giới hạn sau đây còn tùy thuộc vào tình trạng bộ nhớ máy tính của bạn.

Chiều dài tối đa của một dòng mã lệnh : 4095.

Chiều dài tối đa của một chuỗi : 2,147,483,647 kí tự.

Khoảng số thực chấp nhận : 1.7E-308 đến 1.7E+308 với độ chính xác 15 chữ số.

Khoảng số nguyên chấp nhận : số nguyên có dấu 64-bit.

Khoảng số hệ thập lục phân : số nguyên có dấu 32-bit (0X80000000 đến 0X7FFFFFFF).

Mảng : số chiều tối đa là 64 và/hoặc số phần tử là 16 triêu.

Số cấp độ khi gọi đệ quy hàm : 5100 cấp.

Số lượng tối đa các biến được sử dụng tại cùng một thời điểm : không giới hạn.

Số lượng hàm do người dùng định nghĩa : không giới hạn.

Số lượng tối đa cửa sổ GUI mà người dùng có thể tạo : không giới hạn.

Số lượng tối đa của control : 65532.

6. Tôi có thể chọn một ảnh icon cho tập tinh *.au3 sau khi biên dịch hay không ?

Để làm được điều này, bạn phải dùng cửa sổ biên dịch đầy đủ chức năng hơn của AutoIt thay vì phải nhắp chuột phải vào tập tin mã *.au3 sau đó chọn Compile Script. Bạn có thể xem hướng dẫn như sau :

- Mở chương trình Autot2Exe v3 bằng cách vào Start -> All Programs -> AutoIt v3 -> Compile Script to .exe. Giao diện chính của chương trình như sau :

- Nhấn nút Browse tại mục Source (AutoIt .au3 file) để chọn tập tin *.au3 cần biên dịch và tại mục Destination (.exe/.a3x file) để chọn đường dẫn thư mục chứa tập tin *.exe sau khi biên dịch.

- Bây giờ bạn có thể thiết lập icon cho tập tin *.exe sau khi biên dịch bằng cách nhấn nút Browse trong mục Custom Icom (.ico file) và duyệt tìm đến tập tin *.ico của bạn. Bạn có thể sử dụng các icon mẫu của AutoIt trong thư mục Program Files\AutoIt3\Auto2Exe\Icons.

- Một tùy chọn khác cho phép bạn thiết lập là độ nén của tập tin *.exe sau khi biên dịch (đặc biệt nếu bạn có dùng hàm FileInstall trong tập tin *.au3 của bạn). Bạn hãy dùng menu Compression để nén. Cũng như các trình nén tập tin khác, bạn nên chọn Slower thì tốt hơn. Tuy nhiên không có gì đối với cấp độ nén mà bạn chọn, tốc độ giải nén khi chạy tập tin *.exe là như nhau mà thôi.

- Nhấn Convert đế tiến hành biên dịch.

7. Làm thế nào để tôi có thể thoát đoạn mã script của tôi đang chạy bằng một tố hợp phím ?

Thật là dễ dàng. Nếu bạn muốn thoát một đoạn mã script đang chạy khi bạn nhấn một tổ hợp phím thì bạn có thể dùng hàm HotKeySet để thiết lập một tổ hợp phím nóng gán với việc gọi hàm thoát do người dùng định nghĩa. Hàm thoát của người dùng định nghĩa nên chỉ có từ khóa Exit. Ở dưới đây là đoạn code sẽ làm cho đoạn mã script đang chạy thoát đi bằng cách nhấn tổ hợp phím Ctrl – Alt – x.

```
HotKeySet("^!x", "MyExit")
...
...
; Rest of Script
...
...
Func MyExit()
    Exit
EndFunc
```

8. Tại sao tôi gặp lỗi khi dùng dấu ngoặc kép trong khi thiết lập chuỗi cho một biến?

Nếu bạn muốn dùng dấu ngoặc kép bên trong một chuỗi kí tự, bạn phải gõ hai dấu ngoặc kép để thay cho một dấu ngoặc kép. Ví dụ nếu bạn muốn thiết lập một biến chứa chuỗi Le "Vien" Trinh thi bạn có thể làm như sau :

$var = "Le ""Vien"" Trinh"

Hoặc bạn có thể dùng dấu nháy đơn để bao quanh chuỗi như sau

$var = 'Le "Vien" Trinh'

9. Tôi thấy một khung nhỏ như icon nhưng nó không chứa ảnh gì cả trong tập tin trợ giúp của AutoIt dưới phần Example trong phần giới thiệu mỗi hàm ?

Khung nhỏ như icon mà bạn nói đáng lẽ phải là nút Open this Script dùng để mở đoạn mã ví dụ trong phần Example với chương trình SciTE để cho phép bạn có thể chạy thử ví dụ. Vấn đề này xảy ra khi tập tin hhctrl.ocx không được đăng kí với hệ thống hoặc nó bị lỗi. Bạn hãy cố gắng gõ lệnh regsvr32 hhctrl.ocx trong cửa sổ cmd coi có thể giải quyết được không hoặc kiểm tra tập tin trong hệ thống coi nó có tồn tại không.

10. Có gì khác nhau giữa giá trị trả về của hàm và @error ?

Thông thường nếu hàm có một giá trị trả về thì hàm chứng tỏ hàm được gọi thành công. Tuy nhiên nếu hàm trả về có lỗi gì đó thì thông thường chúng sẽ tiến hành gán @error do AutoIt định nghĩa tương ứng với lỗi xảy ra khi hàm được gọi. Điều này sẽ giúp cho người lập trình có thể xác định lỗi xảy ra để có thể có những đoạn code xử lý cho phù hợp.

11. Điều gì xảy ra khi tôi dùng hàm Send () với tham số key là một biến ?

Như bạn đã biết hàm Send() dùng để giúp cho bạn giả lập phím nhấn trên bàn phím. Trong trường hợp bạn cung cấp tham số key là tên một biến mà nội dung của biến đó có chứa các kí tự như ! ^ + {SPACE} thì nó cũng chuyển dịch đến các phím nhấn tương ứng trên bàn phím tùy thuộc bạn thiết lập tham số flag trong hàm như thế nào. Xem lại hàm Send() để hiểu rõ thêm về vấn đề này.

Nói chung bạn có thể dùng một biến chứa các chuỗi quy định phím nhấn trên bàn phím thì bạn có thể dùng biến đó trong tham số key của hàm Send().

Ví dụ : *Sleep(5000)*

$a = "^a"

Send($a); nhấn tổ hợp phím Ctrl - A

12. Trong một số hàm về cửa sổ có hai tham số mà tôi thường gặp như "title" và "text". Nó là gì vậy ?

Hầu hết các cửa sổ đều có thể xác định hoặc để phân biệt với nhau thông qua thanh tiêu đề của cửa sổ hoặc kết hợp giữa tiêu đề và văn bản trong cửa sổ đó. Khi bạn sử dụng công cụ AutoIt Window Info giúp bạn lấy thông tin về tham số title và văn bản trong một cửa sổ bất kì. Các tiêu đề của cửa sổ bạn có thể nhận biết dễ dàng, chẳng hạn như Untitled – Notepad là tiêu đề cửa sổ chương trình Notepad trong Windows. Nếu bạn không cung cấp tham số title và text trong hàm thì cửa sổ hiện đang được kích hoạt sẽ được dùng. Tuy nhiên điều này là không đúng trong trường hợp bạn thiết lập tùy chọn WinTitleMatchMode. Tùy chọn này cho phép bạn thiết lập chỉ dẫn cho AutoIt cách dò tìm cửa sổ tương ứng với tham số title được cho (xem tài liệu AutoIt Help ở mục Function Reference -> Misc. Management -> AutoItSetOption tìm đến tham số WinTitleMatchMode trong bảng Option). Lưu ý là các tham số title và text đều có phân biệt chữ hoa và chữ thường. Bạn phải xác định rõ ràng trước khi cung cấp hai tham số trên. Để tránh khỏi phải cung cấp sai tham số title và text, bạn có thể dùng công cụ AutoIt Window Info (Start -> All Programs -> AutoIt v3 -> AutoIt Windows Info) và nhấn tổ hợp phím Ctrl – C để sao chép các tham số title và text trong cửa sổ mà chương trình cung cấp sau đó dán hai tham số trên vào đoạn mã của bạn. Hầu hết các hàm chức năng liên quan đến cửa sổ trong AutoIt đều có hai tham số title và text, chẳng hạn như hàm WinWaitActive dưới đây. Hàm này có công dụng là tạm dừng đoạn mã đang thực thi và chỉ chạy tiếp khi có một cửa sổ nào đó được kích hoạt.

WinWaitActive("title", ["text"], [timeout])

Tham số title trong hàm là bắt buộc phải có. Tham số text và timeout là hai tham số tùy chọn. Trong một số hàm thì tham số text không phải là tham số tùy chọn. Nếu bạn không muốn sử dụng tham số này, bạn có thể cung cấp nó là "" (chỗi rỗng). Nếu bạn cung cấp tham số chuỗi rỗng thì sẽ cho AutoIt biết rằng bất kì một văn bản nào trong cửa sổ đều là lợp lệ. Thật sự ra bạn nên cung cấp thêm tham số text trong trường hợp hai cửa sổ tạo ra có cùng tiêu đề. Việc cung cấp tham số text cho biết bạn đang muốn dùng cửa sổ nào trong chương trình của mình. Giả sử tôi có hai cửa sổ chương trình Notepad và đánh số nó là cửa sổ 1 và cửa sổ 2. Nếu tôi cung cấp cho các hàm chỉ có tham số title thì làm thế nào AutoIt biết dùng cửa sổ nào trong hàm của mình. Lúc này tham số text sẽ giúp phân biệt hai cửa sổ khác nhau tùy thuộc bạn muốn dùng cửa sổ nào mà thôi. Tham số text là văn bản trong cửa sổ mà AutoIt có thể phát hiện được. Nó thường nằm trong khung chứa văn bản chẳng hạn như hình sau, đó là chuỗi văn bản "Viet Nam que huong toi".

Tuy nhiên tham số text trong hàm cũng có thể cung cấp là văn bản nằm trên các nút nhấn như "Yes", "No", "Next". Các hộp thoại có chứa các nội dung văn bản như "Are you sure you wish to continue ?" hoặc văn bản nằm trên các control hoặc cũng có thể bạn không biết nó có từ đâu.

Điều quan trọng là bạn phải cần biết các tham số title và text mà bạn muốn làm việc với các cửa sổ nào đó.

Bây giờ bạn hãy mở công cụ AutoIt Windows Info và chẳng hạn tôi đang tìm thông tin title và text trong cửa sổ như ở hình trên thì bạn có thể thấy thông tin mà chương trình cung cấp như sau trong thẻ Summary

Bây giờ giả sử tôi muốn dùng hai tham số title và text trong cửa sổ như trên để cung cấp tham số trong hàm WinWaitActive đã nói ở trên thì tôi gõ như sau :

WinWaitActive("Untitled – Notepad", "Viet Nam que huong toi")

13. Tại sao tôi chỉ có thể dùng hàm Run() để thực thi các tập tin có phần mở

rộng *.exe mà thôi. Còn các định dạng khác như *.msi và *.txt thì sao ?

Nói chung chỉ có một số phần mở rộng của tập tin có thể tự chạy được thôi, chẳng hạn như *.exe, *.bat, *.com, *.pif. Những định dạng khác như *.txt và *.msi thường khởi chạy với một tiến trình khác. Chẳng hạn khi bạn nhấn đối chuột vào một tập tin có phẩn mở rộng *.msi thì trong tiến trình của hệ thống sẽ có thêm tiến trình msiexec.exe sẽ giúp thực thi các tập tin *.msi của bạn. Do đó để khởi chạy tập tin *.msi trong AutoIt, bạn có thể sử dụng dòng lệnh sau :

RunWait(@COMSPEC & " /c Start myfile.msi")

Ví dụ : RunWait(@COMSPEC & " /c Start E:\Soft\FRNetUserManual-en.msi") ; khởi chạy tập tin FRNetUserManual-en.msi trong thư mục E:\Soft.

Bạn cũng có thể dùng hàm ShellExcuteWait để thực thi tập tin của bạn. Ví dụ như

ShellExecuteWait("E:\Soft\FRNetUserManual-en.msi")

Ghi chú :

- Hàm RunWait có công dụng thực thi một tập tin và tạm dừng đoạn mã lệnh AutoIt đang chạy. Đoạn mã chương trình sẽ chạy tiếp khi tập tin đó không còn chạy nữa.

- Hàm ShellExcuteWait cũng có công dụng tương tự như hàm RunWait tuy nhiên nó dùng API ShellExecute để khởi chạy tập tin.

Bạn có thể xem thêm hai hàm này trong tài liệu AutoIt Help để biết rõ hơn.

14. Tôi có thể chạy một lệnh DOS trong AutoIt hay không ?

Nếu bạn muốn chạy một lệnh DOS trong AutoIt chẳng hạn như lệnh "Dir" thì bạn cần phải chạy nó thông qua bộ thông dịch lệnh (đó có thể là tập tin command.com hay tập tin cmd.exe tùy thuộc vào hệ điều hành của bạn). Bạn có thể sử dụng macro @Comspec để biết vị trí chính xác của tập tin thông dịch lệnh này. Bạn nên dùng hàm RunWait để thực thi một lệnh DOS trước khi thực thi các lệnh AutoIt sau lệnh RunWait này. Dưới đây là một ví dụ lệnh Dir.

RunWait(@COMSPEC & " /c Dir C:\") ; Chú ý là có một khoảng trắng trước /c.

15. Tại sao tôi không thể in nội dung của một biến thông qua lệnh "My var is $variable" ?

Nếu bạn có một biến chẳng hạn như $msg và bạn dùng lệnh sau đây để thể hiện nội dung của một biến thì nó sẽ không làm việc.

MsgBox(0, "Example", "My variable is $msg")

Nếu bạn dùng lệnh này nó sẽ chỉ hiện thị chuỗi "My variable is $msg". Nếu bạn muốn in nội dung của một biến, bạn phải dùng toán tử & để kết nối giữa chuỗi và biến như sau :

MsgBox(0, "Example", "My variable is " & $msg)

Ghi chú : nếu bạn có nhiều biến và muốn hiển thị nó trong nội dung của một chuỗi, bạn nên dùng hàm StringFormat() để cho tiện. Chẳng hạn nếu bạn muốn chèn nội dung của $var1 đến $var5 trong chuỗi bạn có thể dùng lệnh sau

$msg = StringFormat("Var1 is %s, Var2 is %s, Var3 is %s, Var4 is %s, Var5 is %s", $var1, $var2, $var3, $var4, $var5) ; tham số %s để in theo dạng chuỗi cho biến $var1 đến var5.

MsgBox(0, "Example", $msg)

Hàm StringFormat này tương tự như cú pháp in chuỗi trong C dùng hàm printf.

Chọn trình soạn thảo mã lệnh AutoIt nào ?

Các tập tin AutoIt (*.au3) là những tập tin dạng văn bản thông thường và do đó bạn có thể chỉnh sửa nó bằng một trình soạn thảo văn bản bất kì. Tuy nhiên cũng có nhiều ứng

dụng miễn phí hoặc dạng shareware có nhiều chức năng hơn trong việc soạn thảo mã nguồn cũng như hỗ trợ tô sáng cú pháp làm cho nó nổi bật hơn. Do đó việc dùng các công cụ đó để lập trình dễ dàng hơn cũng như tránh gây nhầm lẫn.

Trình soạn thảo mặc định dành cho phần lớn người sử dụng AutoIT chính là SciTE với tính năng làm nổi bật cú pháp ngôn ngữ cũng như tích hợp các công cụ khác như công cụ kiểm tra cú pháp (syntax checking). Phiên bản SciTE có trong bản cài đặt của AutoIT tuy nhiên nó chỉ là phiên bản "lite" mà thôi. Bạn có thể vào trang http://www.autoitscript.com/site/autoit-script-editor/ để tải phiên bản đầy đủ hơn của trình soạn thảo SciTE này.

Dưới đây là danh sách các trình soạn thảo cho AutoIt được khuyến khích sử dụng

- TextPad (http://www.textpad.com/)
- Crimson Editor (miễn phí - http://www.crimsoneditor.com/)
- Source Edit (miễn phí - http://www.sourceedit.com/)
- UltraEdit (http://www.ultraedit.com/)