**MFC programming**

**CSM-VNG**

* **HWND**
* **MFC Classes**
* **Window messages**
* **CWinApp**
* **CWnd**
* **CDialog**

**Author : AnLNT**

Mục lục

[I. HWND 2](#_Toc329240292)

[1. Window là gì? 2](#_Toc329240293)

[2. Handle là gì? 2](#_Toc329240294)

[3. HWND là gì? 2](#_Toc329240295)

[II. MFC Classes 2](#_Toc329240296)

[III. Window messages 2](#_Toc329240297)

[1. SendMessage 2](#_Toc329240298)

[2. PostMessage 3](#_Toc329240299)

[IV. CWinAPP 3](#_Toc329240300)

[1. InitInstance 3](#_Toc329240301)

[2. ExitInstace 3](#_Toc329240302)

[V. CWnd 4](#_Toc329240303)

[WindowProc 4](#_Toc329240304)

[VI. CDialog 5](#_Toc329240305)

[1. DoMoal 5](#_Toc329240306)

[2. OnInitDialog 5](#_Toc329240307)

[3. OnDestroy 6](#_Toc329240308)

# HWND

## Window là gì?

* Trong ứng dụng giao diện đồ họa trên hệ điều hành Windows thì window là vùng hình chữ nhật của màn hình. Window cho phép ứng dụng hiển thị output đến người dùng và nhận input từ người dùng.
* Window bao gồm : cửa sổ, dialog, button,…

## Handle là gì?

* Là một số nguyên 32 bit không dấu do HĐH tạo ra dùng làm định danh cho các đối tượng (window, file, vùng nhớ, menu,…).
* Handle có vẻ giống con trỏ

## HWND là gì?

* Là một handle của một window
* Mọi thao tác chúng ta muốn thực hiện với window (move, show, destroy, chỉnh sửa stylel,…) đều thực hiện thông qua HWND của cửa sổ đó.

# MFC Classes

# Window messages

## SendMessage

**LRESULT SendMessage(**

**UINT message,**

**WPARAM wParam = 0,**

**LPARAM lParam = 0**

**);**

* **Mô tả**

Phương thức này gửi một thông điệp cụ thể đến một cửa sổ.

* **Tham số**
* Message : ID của thông điệp được gửi
* wParm, lParam : các thông tin đi kèm theo thông điệp
* **Giá trị trả về**

Là kết quả của quá trình xử lý thông điệp và kết quả này là tùy thuộc vào loại thông điệp được gửi

* **Lưu ý**

Phương thức SendMessage gọi trực tiếp hàm xử lý thông điệp của cửa sổ và không return cho đến khi nào thông điệp đã được xử lý xong. Trái với phương thức PostMessage là phương thức PostMessage đặt thông điệp vào hàng đợi thông điệp của cửa sổ và return ngay lập tức

## PostMessage

**BOOL PostMessage(**

**UINT message,**

**WPARAM wParam = 0,**

**LPARAM lParam = 0**

**);**

* **Mô tả**

Phương thức này đặt thông điệp vào hàng đợi thông điệp của cửa sổ và return ngay lập tức mà không cần quan tâm đến quá trình xử lý thông điệp đó

* **Tham số**
* Message : ID của thông điệp được đặt vào hàng đợi
* wParm, lParam : các thông tin đi kèm theo thông điệp
* **Giá trị trả về**

Khác 0 nếu như thông điệp được đặt vào hàng đợi thành công. Ngược lại trả về 0

* **Lưu ý**

# CWinAPP

## InitInstance

**virtual** **BOOL** **InitInstance(** **);**

* **Mô tả**

Phương thức này được gọi ngay khi ứng dụng hoặc DLL được khởi tạo

* **Giá trị trả về**
* Nếu khởi tạo thành công thì trả về 1. Ngược lại trả về 0
* Nếu trả về 0 thì ứng dụng sẽ bị shut down
* **Lưu ý**

Thường thì trong phương thức InitInstance này sẽ thực hiện việc khởi tạo cửa sổ chính của ứng dụng.

## ExitInstace

**virtual** **int** **ExitInstance(** **);**

* **Mô tả**
* Phương thức này được gọi khi ứng dụng hoặc dll bị exit
* Đây là nơi tốt nhất để dọn dẹp hệ thống, thu hồi các tài nguyên đã được cấp phát
* **Giá trị trả về**
* Dùng làm mã thoát của ứng dụng. Thường là 0 muốn nói rằng ứng dụng kết thúc một cách bình thường. Khác 0 là ứng dụng thoát do bị lỗi gì đó.
* Giá trị này cũng chính là giá trị trả về của hàm winmain
* **Lưu ý**

Mặc định thì phương thức ExitInstace này sẽ thực hiện các thủ tục dọn dẹp các tài nguyên được yêu cầu cấp phát bởi MFC Framework. Do đó khi override lại phương thức này thì phải dùng lớp cha gọi thực hiện phương thức ExitInstace trước.

# CWnd

## WindowProc

**virtual LRESULT WindowProc(**

**UINT message,**

**WPARAM wParam,**

**LPARAM lParam**

**);**

* **Mô tả**

Thay thế hàm xử lý thông điệp mặc định của cửa số bằng hàm xử lý thông điệp do chính mình cài đặt

* **Tham số**
* Message : ID của thông điệp được gửi đến
* wParam, lParam : các thông tin đi kèm theo thông điệp
* **Giá trị trả về**

Phụ thuộc vào thông điệp được gửi

* **Lưu ý**

Ở cuối phương thức này nên gọi hàm xử lý thông điệp của lớp cha để có một số thông điệp mà ta không muốn xử lý thì sẽ được giao cho lớp cửa sổ cha xử lý. Nếu ta không gọi hàm xử lý thông điệp của cửa sổ cha thì rất nhiều thông điệp gửi đến cửa sổ sẽ không được xử lý. Như thế rất nguy hiểm. Ví dụ như khi cửa sổ được tạo thì Hệ điều hành gửi thông điệp WM\_CREATE đến mà không có ai xử lý thì sẽ không có cửa sổ.

LRESULT CMainWindow::WindowProc(UINT message, WPARAM wParam, LPARAM lParam ) {

//su ly mot so thong diep

return CFrameWnd::WindowProc(message, wParam, lParam);

}

# CDialog

## DoMoal

**DoModal();**

* **Mô tả**
* Gọi phương thức này để hiển thị **modal** dialog box.
* Phương thức này sau khi được gọi sẽ không return cho đến khi nào người dùng đóng dialog box này.
* **Giá trị trả về**
* -1 nếu không hiển thị được dialog box
* Các trường hợp còn lại là tham số được truyền vào phương thức EndDialog khi đóng dialog box

## OnInitDialog

**virtual BOOL OnInitDialog( );**

* **Mô tả**
* Phương thức này được gọi khi dialog box nhận được thông điệp WM\_INITDIALOG
* Thông điệp WM\_INITDIALOG được gửi sau khi các control trên dialog đã được khởi tạo và trước khi dialog được hiển thị với người dùng
* OnInitDialog là nơi ta sẽ thực hiện các công việc cần thiết để dialog box sẵn sàng hiển thị với người dùng (vd : set focus cho control, check vào một vài checkbox nào đó, chỉ định text hiển thị trong các textbox,..). Tại thời điểm gọi OnInitDialog thì dialog đã có trong bộ nhớ nhưng nó chưa được hiển thị với người dùng
* **Giá trị trả về**

Để cho hệ điều hành biết sẽ set focus vào control nào trên dialog box

* TRUE : hệ điều hành sẽ luôn luôn set focus vào phần tử đầu tiên theo thứ tự tab order.
* FALSE : thì quyền setfocus được giao cho lập trình viên. Nếu lập trình viên không chỉ định thì mặc định control đầu tiên theo thứ tự tab order sẽ được set focus
* **Lưu ý**
* MSDN và nhiều tài liệu khuyên khi override phương thức OnInitDialog thì phải dùng lớp cha gọi phương thức OnInitDialog.
* Em thử không dùng lớp cha gọi phương thức OnInitDialog mà chương trình vẫn không có lỗi gì

## OnDestroy

**virtual VOID OnDestroy( );**

* **Mô tả**

Phương thức này được gọi nhằm thông báo rằng cửa sổ đang bị hủy

* **Lưu ý**
* OnDestroy được gọi sau khi cửa sổ đã bị xóa khỏi màn hình
* Sau khi cửa sổ bị hủy thì các cửa sổ(control) con của cửa sổ đó cũng sẽ bị hủy theo