<https://msdn.microsoft.com/en-us/library/aa363215%28VS.85%29.aspx>

<https://msdn.microsoft.com/en-us/library/aa363431.aspx>

https://msdn.microsoft.com/en-us/library/aa363432.aspx

<http://stackoverflow.com/questions/2184123/detect-removable-drive-e-g-usb-flash-drive-c-c>

C++

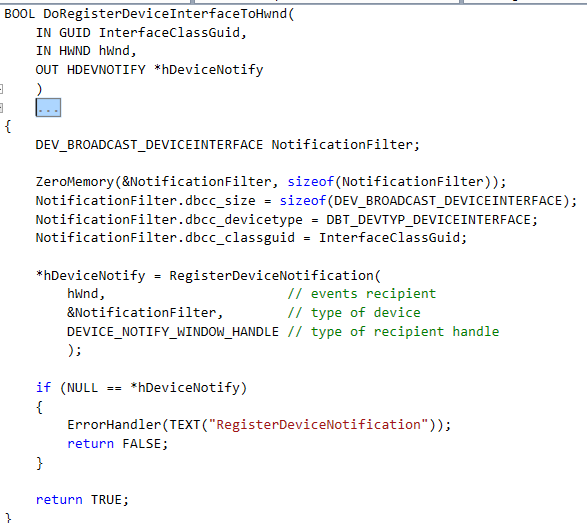
Ứng dụng gửi thông báo sự kiện bằng hàm BroadcastSystemMessage. Mọi ứng dụng tầng trên có thể nhận thông báo bằng cách xử lý tin nhắn WM\_DEVICECHANGE. Ứng dụn có thể dùng hàm RegisterDeviceNotification để đăng ký nhận thông báo về các thiết bị.

Service có thể dùng hàm RegisterDeviceNotification để đăng ký nhận thông báo thiết bị. Nếu một service chỉ định một window handle trong hRecipient parameter, thông báo được gửi đến window procedure. Nếu hRecipient là một dịch vụ kiểm soát trạng thái, **SERVICE\_CONTROL\_DEVICEEVENT** thông báo được gửi đến cho dịch vụ điều khiển xử lý.

Sự kiện **DBT\_DEVICEARRIVAL** và **DBT\_DEVICEREMOVECOMPLETE** được tự động phát thông báo cho tất cả các ứng dụng tầng trên cho cổng các thiết bị. Hơn thế nữa, việc này không cần thiết gọi RegisterDeviceNotification cho các cổng, và hàm thất bại nếu dbch\_devicetype là DBT\_DEVTYP\_VOLUME. Thiết bị OEM-defined không sử dụng trực tiếp bởi hệ thống, do đó cá chức năng thất bại nếu dbch\_devicetype là **DBT\_DEVTYP\_OEM**.

Khai báo **GUID** cho tất cả các **USB serial host PnP drivers** (có thể thay thế bằng bất kỳ GUID hợp lệ khác)





Hàm **DoRegisterDeviceInterfaceToHwnd**(IN GUID InterfaceClassGuid, IN HWND hWnd, OUT HDEVNOTIFY \*hDeviceNotify) có chức năng đăng ký cho một Hwnd để nhận thông báo thay đổi giao tiếp của các thiết bị có GUID được chỉ định ở **WcenusbshGUID**. Bên trong hàm **DoRegisterDeviceInterfaceToHwnd**() gọi hàm:

HDEVNOTIFY WINAPI RegisterDeviceNotification(

\_In\_  HANDLE hRecipient,

\_In\_  LPVOID NotificationFilter,

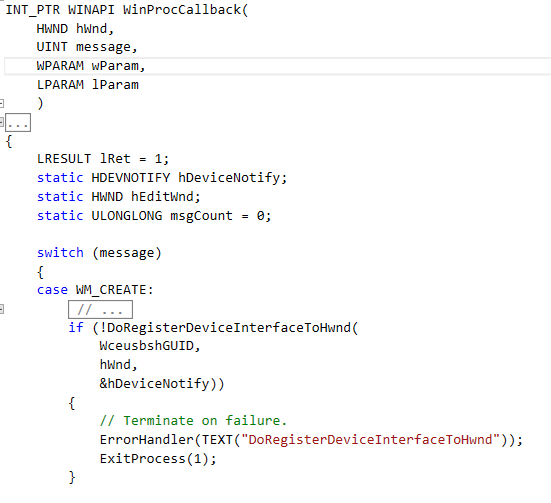
\_In\_  DWORD Flags

);

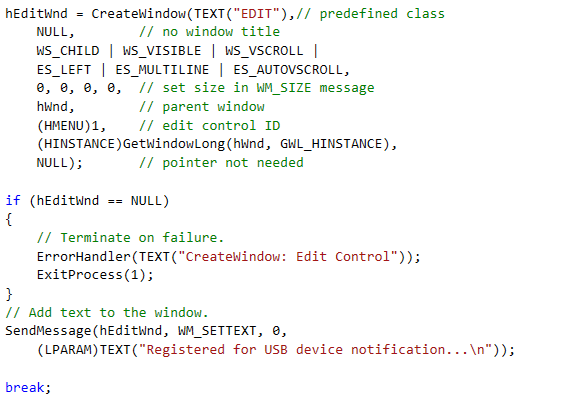
* *hRecipient* [in]: một handle của window hoặc service sẽ nhận các sự kiện của thiết bị được chỉ định trong tham số *NotificationFilter*. Những window handle tương tự có thể sử dụng trong multiple calls tới **RegisterDeviceNotification**. Service có thể xác định cả windows handle và service status handle.
* *NotificationFilter* [in]: Một con trỏ tới block dữ liệu xác định kiểu thiết bị mà thông báo nên được gửi. Block này luôn bắt đầu với cấu trúc **DEV\_BROADCAST\_HDR**. Dữ liệu theo sau header này phụ thuộc vào giá trị của **dbch\_devicetype**, nó có thể là **DBT\_DEVTYP\_DEVICEINTERFACE** hoặc **DBT\_DEVTYP\_HANDLE**.
* *Flags* [in]: Tham số này có thể là một trong các giá trị sau:

|  |  |
| --- | --- |
| **DEVICE\_NOTIFY\_WINDOW\_HANDLE**  0x00000000 | Tham số *hRecipient* là một window handle. |
| **DEVICE\_NOTIFY\_SERVICE\_HANDLE**  0x00000001 | Tham số *hRecipient* là một service status handle. |

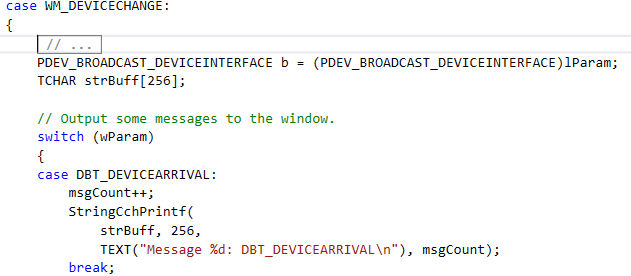
Hàm WindProcCallback(…) tiến hành đăng ký nhận thông báo trong WM\_CREATE. Gọi hàm DoRegisterDeviceInterfaceToHwnd() để đăng ký một lần duy nhất khi khởi tạo



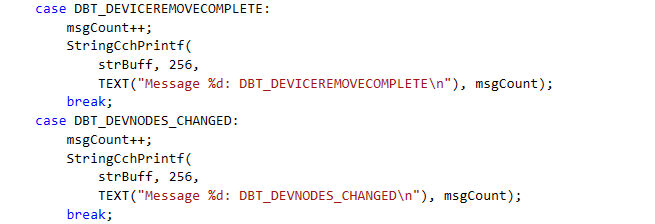
Cuối cùng, tạo cửa sổ giao diện trong WM\_CREATE



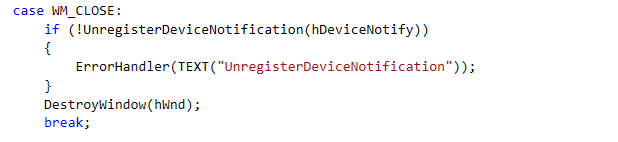
Xử lý trong WM\_DEVICECHANGE



Xuất ra màn hình thông báo khi máy tính nhận thiết bị



Tương tự đối với các thông báo khác nhận được, chương trình xuất ra câu thông báo tương ứng.



Khi đóng ứng dụng, thực hiện thao tác hủy đăng ký nhận thông báo với hàm

BOOL WINAPI UnregisterDeviceNotification(

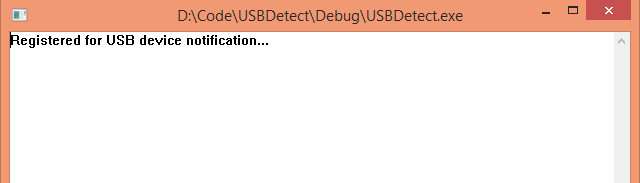
\_In\_  HDEVNOTIFY Handle

);

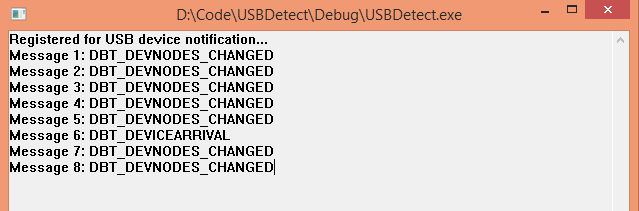
*Handle* [in]: được trả về khi đăng ký thành công **RegisterDeviceNotification**()

Demo

Giao diện khi khởi động



Khi cắm 1 USB vào máy tính



Khi rút USB ra khỏi máy tính

