
Subject: Re: Using Pen with U++

Posted by [mirek](#) on Tue, 09 Mar 2021 16:48:14 GMT

[View Forum Message](#) <> [Reply to Message](#)

Tom1 wrote on Tue, 09 March 2021 16:56How about this approach using

EnableMouseInPointer() as follows?

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

```
    GuiLock __;
```

```
    eventid++;
```

```
// LLOG("Ctrl::WindowProc(" << message << ") in " << ::Name(this) << ", focus " << (void
*)::GetFocus());
```

```
Ptr<Ctrl> _this = this;
```

```
HWND hwnd = GetHWND();
```

```
ONCELOCK {
```

```
    EnableMouseInPointer(true); // #1 - enable mouse in WM_POINTER* messages
```

```
};
```

```
switch(message) {
```

```
case WM_POINTERDOWN:
```

```
case WM_POINTERUPDATE:
```

```
case WM_POINTERUP:
```

```
{
```

```
    POINT p = Point(lParam);
```

```
    ScreenToClient(hwnd, &p);
```

```
    pen = false;
```

```
    pen_pressure = pen_rotation = Null;
```

```
    pen_tilt = Null;
```

```
    pen_eraser = false;
```

```
    pen_barrel = false;
```

```
    pen_inverted = false;
```

```
static BOOL (WINAPI *GetPointerType)(UINT32 pointerId, POINTER_INPUT_TYPE
*pointerType);
```

```
static BOOL (WINAPI *GetPointerInfo)(UINT32 pointerId, POINTER_INFO *pointerInfo);
```

```
static BOOL (WINAPI *GetPointerPenInfo)(UINT32 pointerId, POINTER_PEN_INFO *penInfo);
```

```
static BOOL (WINAPI *GetPointerTouchInfo)(UINT32 pointerId, POINTER_TOUCH_INFO
*touchInfo);
```

```
ONCELOCK {
```

```
    DllFn(GetPointerType, "User32.dll", "GetPointerType");
```

```
    DllFn(GetPointerInfo, "User32.dll", "GetPointerInfo");
```

```
    DllFn(GetPointerPenInfo, "User32.dll", "GetPointerPenInfo");
```

```
    DllFn(GetPointerTouchInfo, "User32.dll", "GetPointerTouchInfo");
```

```
};
```

```
POINTER_INPUT_TYPE pointerType;
```

```
UINT32 pointerId = GET_POINTERID_WPARAM(wParam);
if(GetPointerType && GetPointerPenInfo && GetPointerType(pointerId, &pointerType)) {
    switch(pointerType){
    case PT_PEN:{
        POINTER_PEN_INFO ppi;
        if(GetPointerPenInfo(pointerId, &ppi)) {
            pen = true;
            if(ppi.penFlags & PEN_FLAG_BARREL)
                pen_barrel = true;
            if(ppi.penFlags & PEN_FLAG_INVERTED)
                pen_inverted = true;
            if(ppi.penFlags & PEN_FLAG_ERASER)
                pen_eraser = true;
            if(ppi.penMask & PEN_MASK_PRESSURE)
                pen_pressure = ppi.pressure / 1024.0;
            if(ppi.penMask & PEN_MASK_ROTATION)
                pen_rotation = ppi.rotation * M_2PI / 360;
            if(ppi.penMask & PEN_MASK_TILT_X)
                pen_tilt.x = ppi.tiltX * M_2PI / 360;
            if(ppi.penMask & PEN_MASK_TILT_Y)
                pen_tilt.y = ppi.tiltY * M_2PI / 360;

        }
        break;
    }
    case PT_TOUCH:{
        POINTER_TOUCH_INFO pti;
        if(GetPointerTouchInfo(pointerId, &pti)) {
            // Add something touch specific here some day maybe...
        }
        break;
    }
    default:{
        POINTER_INFO pi;
        if(GetPointerInfo(pointerId, &pi)) {
        }
        break;
    }
}

if(!_this) switch(message){
case WM_POINTERDOWN:
    ClickActivateWnd();
    DoMouse(LEFTDOWN, Point(p), 0);
    PostInput();
```

```

    break;
case WM_POINTERUP:
    DoMouse(LEFTUP, Point(p), 0);
    PostInput();
    break;
case WM_POINTERUPDATE:
    DoMouse(MOUSEMOVE, Point(p));
    DoCursorShape();
    break;
}

}
}
break;
case WM_POINTERLEAVE:
    pen = false;
    break;

```

Further more, this requires disabling WM_LEFTUP, WM_LEFTDOWN and WM_MOUSEMOVE as they now come in as WM_POINTER* messages.

IMPORTANT WARNING: I have not replicated their function completely in WM_POINTER above as I do not quite understand all the finer details there. Also, I'm worried about breaking everything, so I'll just throw this in 'as is' for you to consider...

Oh and yes, this fixes the 2 cm straight line start issue... :)

Best regards,

Tom

EDIT: EnableMouseInPointer() is available on Windows8 and above only, so it needs to be loaded in a similar way as the rest of the Pointer -functions.

Looks good. Will try tommorow.
