

---

Subject: Re: Using Pen with U++  
Posted by [Tom1](#) on Fri, 12 Mar 2021 19:27:02 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Mirek,

Can you test this:

```
switch(message) {
case WM_POINTERDOWN:
case WM_POINTERUPDATE:
case WM_POINTERUP:
{

POINT p = Point((LONG)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 *EnableMouseInPointer)(BOOL fEnable);
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);
static BOOL (WINAPI *GetPointerPenInfoHistory)(UINT32 pointerId, UINT32 *entriesCount,
POINTER_PEN_INFO *penInfo);

ONCELOCK {
DllFn(GetPointerType, "User32.dll", "GetPointerType");
DllFn(GetPointerInfo, "User32.dll", "GetPointerInfo");
DllFn(GetPointerPenInfo, "User32.dll", "GetPointerPenInfo");
DllFn(GetPointerTouchInfo, "User32.dll", "GetPointerTouchInfo");

DllFn(GetPointerPenInfoHistory, "User32.dll", "GetPointerPenInfoHistory");
};

POINTER_INPUT_TYPE pointerType;

UINT32 pointerId = GET_POINTERID_WPARAM(wParam);
if(GetPointerType && GetPointerType(pointerId, &pointerType)) {
```

```

switch(pointerType){
case PT_PEN:{

    UINT32 hc=256;
    POINTER_PEN_INFO ppit[hc];
    if(message==WM_POINTERUPDATE && GetPointerPenInfoHistory &&
GetPointerPenInfoHistory(pointerId, &hc, ppit)) {
    for(int i=hc-1;i>=0;i--){
        pen = true;
        if(ppit[i].penFlags & PEN_FLAG_BARREL)
            pen_barrel = true;
        if(ppit[i].penFlags & PEN_FLAG_INVERTED)
            pen_inverted = true;
        if(ppit[i].penFlags & PEN_FLAG_ERASER)
            pen_eraser = true;
        if(ppit[i].penMask & PEN_MASK_PRESSURE)
            pen_pressure = ppit[i].pressure / 1024.0;
        if(ppit[i].penMask & PEN_MASK_ROTATION)
            pen_rotation = ppit[i].rotation * M_2PI / 360;
        if(ppit[i].penMask & PEN_MASK_TILT_X)
            pen_tilt.x = ppit[i].tiltX * M_2PI / 360;
        if(ppit[i].penMask & PEN_MASK_TILT_Y)
            pen_tilt.y = ppit[i].tiltY * M_2PI / 360;

        POINT hp = ppit[i].pointerInfo.ptPixelLocation;
        ScreenToClient(hwnd, &hp);

        if(ignoreclick) {
            EndIgnore();
            return 0L;
        }
        if(!_this) DoMouse(MOUSEMOVE, Point(hp));
        DoCursorShape();

    }
    return 0L;
}

POINTER_PEN_INFO ppi;
if(GetPointerPenInfo && 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 && GetPointerTouchInfo(pointerId, &pti)) {
        // Add something touch specific here some day maybe...
    }
    break;
}
}

switch(message){
case WM_POINTERDOWN:
    ClickActivateWnd();
    if(ignoreclick) return 0L;
    DoMouse(LEFTDOWN, Point(p), 0);
    if(!_this) PostInput();
    break;
case WM_POINTERUP:
    if(ignoreclick) EndIgnore();
    else DoMouse(LEFTUP, Point(p), 0);
    if(!_this) PostInput();
    break;
case WM_POINTERUPDATE:
    if(ignoreclick) {
        EndIgnore();
        return 0L;
    }
    if(!_this) DoMouse(MOUSEMOVE, Point(p));
    DoCursorShape();
    break;

}
return 0L;
}
}
break;
case WM_POINTERLEAVE:
    pen = false;
    break;

```

No blocking of original WM\_MOUSEMOVE, etc. required at all.

For some reason, with this code I do not get those WM\_MOUSEMOVE messages for Pen, but only for mouse. Similarly, WM\_POINTER... messages only appear for Pen.

Right clicks and double clicks work for mouse as before. So this is closest to a working solution so far.

Best regards,

Tom

---