Subject: Re: Using Pen with U++
Posted by Tom1 on Mon, 22 Mar 2021 09:35:04 GMT
View Forum Message <> Reply to Message

Hi,

Here's an improved testcase code for easier callback monitoring:

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

#define LLOG(x)  DLOG(x)

bool moved=false;

class PointerEvent{
public:
 Point p;
 int  row;
 String label;

 PointerEvent(Point p_, String label_, int row_){
  if(moved){
   moved=false;
   LLOG("...");
  }
  p=p_;
  label=label_;
  row=row_;
  LLOG(label << ": " << p);
 }
};

struct MyApp : TopWindow {
 Array<PointerEvent> elist;

 Vector<Vector<Tuple<double, Pointf>>> drawing;
 Vector<Vector<Tuple<double, Pointf>>> rdrawing;

 bool rdown;
 bool ldown;

 Point p1,p2; // Tracker test variables
 Rect trect;

 MyApp(){
  rdown=false;
```

```
 ldown=false;

 moved=false;

 trect=Null;
 p1=p2=Null;

 EnablePenSupport();
}

virtual void RightDown(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenRightDown":"MouseRightDown",0));
 rdown=true;
 rdrawing.Add().Add(MakeTuple(IsPointerPen()?GetPenPressure():0.1, p));
 Refresh();
}

virtual void RightHold(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenRightHold":"MouseRightHold",1));
 Refresh();
}

virtual void RightDrag(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenRightDrag":"MouseRightDrag",-1));
 Refresh();
}

virtual void RightDouble(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenRightDouble":"MouseRightDouble",-2));
 Refresh();
}

virtual void RightTriple(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenRightTriple":"MouseRightTriple",-3));
 Refresh();
}

virtual void RightUp(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenRightUp":"MouseRightUp",-1));
 rdown=false;
 Refresh();
}

virtual void LeftDown(Point p, dword keyflags){
 if(keyflags&K_SHIFT){
  RectTracker tracker(*this);
  tracker.sync= [=](Rect r) {  };
  tracker.MinSize(Size(-100000,-100000));
```

```
  trect=tracker.Track(Rect(p,p));
 }
 else if(keyflags&K_CTRL){
  RectTracker tracker(*this);
  p1=p;
  p2=tracker.TrackLine(p.x,p.y);
 }
 else{
  elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenLeftDown":"MouseLeftDown",0));
  ldown=true;
  drawing.Add().Add(MakeTuple(IsPointerPen()?GetPenPressure():0.1, p));
 }
 Refresh();
}

virtual void LeftHold(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenLeftHold":"MouseLeftHold",1));
 Refresh();
}

virtual void LeftDrag(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenLeftDrag":"MouseLeftDrag",-1));
 Refresh();
}

virtual void LeftDouble(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenLeftDouble":"MouseLeftDouble",-2));
 Refresh();
}

virtual void LeftTriple(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenLeftTriple":"MouseLeftTriple",-3));
 Refresh();
}

virtual void LeftUp(Point p, dword){
 elist.Insert(0,PointerEvent(p,IsPointerPen()?"PenLeftUp":"MouseLeftUp",-1));
 ldown=false;
 Refresh();
}

Vector<String> report;

virtual void MouseMove(Point p, dword keyflags) {
 moved = true;

 if((ldown && drawing.GetCount()) || (rdown && rdrawing.GetCount())) {
  if(rdown){
```

```
    if(!IsPenHistoryEvent()) rdrawing.Top().Add(MakeTuple(IsPointerPen()?GetPenPressure():0.1,
p));
   }
   else if(ldown) drawing.Top().Add(MakeTuple(IsPointerPen()?GetPenPressure():0.1, p));
  }

  report.Clear();
  report.Add() << AsString(p);
  report.Add() << "Pen: " << IsPointerPen();
  report.Add() << "Pressure: " << GetPenPressure();
  report.Add() << "Rotation: " << GetPenRotation();
  report.Add() << "Tilt: " << GetPenTilt();
  report.Add() << "Barrel: " << IsPenBarrelPressed();
  report.Add() << "Inverted: " << IsPenInverted();
  report.Add() << "Eraser: " << IsPenEraserPressed();

  Refresh();
 }

 virtual void Paint(Draw& w0){
  DrawPainter w(w0, GetSize());
  w.Clear(SColorPaper());

  w.LineCap(LINECAP_ROUND);

  for(const auto& stroke : drawing)
   if(stroke.GetCount())
    for(int i = 0; i < stroke.GetCount() - 1; i++) {
     w.Move(stroke[i].b);
     w.Line(stroke[i + 1].b);
     w.Stroke(DPI(20) * stroke[i].a, LtBlue());
    }

  for(const auto& stroke : rdrawing)
   if(stroke.GetCount())
    for(int i = 0; i < stroke.GetCount() - 1; i++) {
     w.Move(stroke[i].b);
     w.Line(stroke[i + 1].b);
     w.Stroke(DPI(20) * stroke[i].a, Black());
    }

  if(!IsNull(trect)) w.RectPath(trect).Stroke(2.0,Red());
  if(!IsNull(p2)) w.Move(p1).Line(p2).Stroke(2.0,Green());

  int fcy = GetStdFontCy();
  int y = 10;
  auto Text = [&] (const String& text) {
   w.DrawText(10, y, text);
```

```
   y += fcy;
  };

  Text("1. Test Clicks, Holds, Drags, DoubleClicks and TripleClicks using Mouse with both Left and
Right buttons.");
  Text("2. Test Clicks, Holds, Drags, DoubleClicks and TripleClicks using Pen without and with
Barrel button.");
  Text("3. Test RectTracker with Ctrl+LeftDrag and Shift+LeftDrag using Mouse and then Pen.");

  for(int i=0;i<report.GetCount();i++) Text(report[i]);

  elist.Trim(min(15,elist.GetCount()));
  for(int i=0;i<elist.GetCount();i++) w.DrawText(elist[i].p.x, elist[i].p.y + elist[i].row*GetStdFontCy(),
elist[i].label);
 }
};

GUI_APP_MAIN
{
 PromptOK("Please tap OK with Pen to verify button operation!");
 MyApp().Run();
}
```

Best regards,

Tom