Simulating Pointer Events

Overview

This snippet shows how to Simulate Pointer events in a Control.

S60 5th Edition devices are Touch UI (Pointer) Devices. Pointer events are handled by the controls. A control should implement the function **HandlePointerEventL()** to be able to handle pointer events. The HandlePointerEventL() function is called by the framework whenever a pointer event occurs within the control.

There are 10 kind of events listed in S60 platform, but we are only concerned with the following four events.

```
TPointerEvent::EButton1Down  /** Button 1 or pen down. */
TPointerEvent::EButton1Up    /** Button 1 or pen up. */

/** This Event is only received after EButton1Down Event. */
TPointerEvent::EDrag

/** These events are only received when button 1 is up and the XY input mode is not pen.
*/
TPointerEvent::EMove
```

Preconditions

Here we assume that we already have a working code for UI based Application.

MMP file

The following capabilities and libraries are required:

```
CAPABILITY None
```

```
LIBRARY cone.lib
```

```
#include <COECNTRL.H>
```

Simulating Pen Down Event.

```
TPointerEvent pointerEvent;
pointerEvent.iType = TPointerEvent::EButton1Down;
HandlePointerEventL( pointerEvent );
```

Simulating Pen Up Event.

```
TPointerEvent pointerEvent;
pointerEvent.iType = TPointerEvent::EButton1Up;
HandlePointerEventL( pointerEvent );
```

Simulating Pen Drag Event.

(C) Copyright Nokia 2014. All rights reserved.

```
TPointerEvent pointerEvent;
pointerEvent.iType = TPointerEvent::EDrag;
HandlePointerEventL( pointerEvent );
```

Simulating Pen Move Event.

```
TPointerEvent pointerEvent;
pointerEvent.iType = TPointerEvent::EMove;
HandlePointerEventL( pointerEvent );
```

Simulating Pointer Events for a specific Control

Note that when a Control is a **Compound Control**, and if you want to simulate Pointer Events for a **Component Control**, then you have to get the two-dimensional Point **Position** in order to enable that pointer event to be processed successfully, because all the **pointer events are ignored** if the pointer events are **not inside** the **control's window**.

For example, If I want to produce/simulate Pen Down Event for Edwin and then I want **Edwin->HandlePointerEventL()** to handle/process it correctly, then i have to first get the location within the edwin window.

```
// Getting the cursor position
TInt edwinCurPos = iEdwin->CursorPos();
// Get the Text View
CTextView* textView = iEdwin->TextView();
TPoint textViewPos;
// Gets the window coordinates of the Cursor position.
textView->DocPosToXyPosL(edwinCurPos, textViewPos);

// Then simulate Pen Down Event for Edwin.
TPointerEvent pointerEvent;
pointerEvent.iPosition = textViewPos;
pointerEvent.iType = TPointerEvent::EButton1Down;
// Now Pen Down Event will be handled correctly by Edwin.
iEdwin->HandlePointerEventL( pointerEvent );
```

Postconditions

Dummy Pointer Events will be passed to Control.

(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.