How to Enable/Disable Drag and Move Events in a Control

This snippet shows how to Enable or Disable Drag and Move Pointer events in a Symbian C++ UI Control.

Overview

By default, pointer drag events and pointer move events are not delivered to controls. The default behaviour when a window is created is that move, drag, enter and exit events are filtered out and not delivered to the client. In order for the control to receive drag events occurring in its window, we need to call `EnableDragEvents()` in the control's construction routine. Having done that, the control will receive drag events as calls to its `HandlePointerEventL()` virtual function.

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>
```

Enabling Drag and Move Events

```
void CSomeAppView::ConstructL(const TRect& aRect)
    {
    // Create a window for the container
    CreateWindowL();
    .
    . .
    . . .
    // Enable Drag Events.
    EnableDragEvents();

    // OR can also be done with
    Window().PointerFilter(EPointerFilterDrag, 0);

    .. ..
    }
```

Handling Drag and Move Events

Control receive's drag and move events in its **HandlePointerEventL()** virtual function, hence whatever processing is required for Drag and Move events can be done inside this function.

```
void CSomeAppView::HandlePointerEventL(const TPointerEvent& aPointerEvent)
{
 switch (aPointerEvent.iType)
 {
  .
  . .
  . . .
```


(C) Copyright Nokia 2014. All rights reserved.

```
   case TPointerEvent::EDrag:
      {
      // Handle drag event here.
      break;
      }
   case TPointerEvent::EMove:
      {
      // Handle move event here.
      break;
      }
   default:
      {
      // do something
      break;
      }
 }
```

Disabling Drag and Move Events

The control framework does not provide a function to disable drag events at a later time, but this can be done via the Window Server API, by calling `Window()->PointerFilter()`.

```
// A 1 bit causes the corresponding event to be filtered out, a 0 bit lets through the
corresponding event.
Window().PointerFilter(EPointerFilterDrag, 1);
```

Postconditions

Drag and Move Pointer Events will be Enabled/Disabled.

(C) Copyright Nokia 2014. All rights reserved.