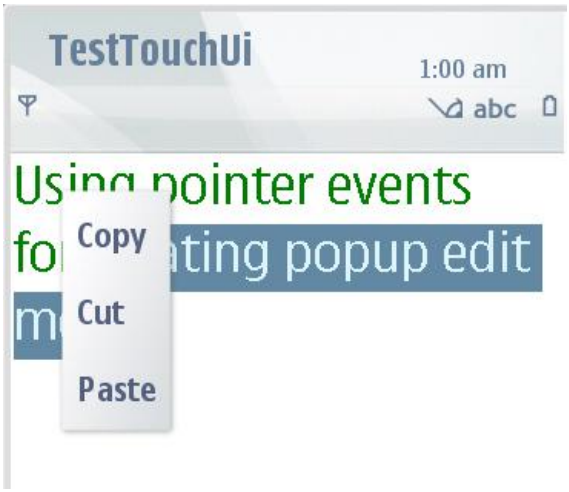
Using pointer events for creating popup edit menu

ID		**Creation date**	October 22, 2008
Platform	S60 5th Edition	**Tested on devices**	
Category	Symbian C++	**Subcategory**	

Keywords (APIs, classes, methods, functions): CCoeControl::HandlePointerEventL(), TPointerEvent::EButton1Up, TPointerEvent::EDrag, CEikEdwin::SelectionLength(), CAknStylusPopUpMenu, CAknLongTapDetector

Overview

Symbian CCoeControl class implementation provides virtual HandlePointerEventL() method for handling pointer event like pointer down/up and dragging as well. Thus class derived from CCoeControl could implement this function just for the case of customization pointer events handling. S60 5th edition provides extended support of touch ui stuff on emulator and real devices. That is why such functionality could and should be used for enriching user interecting expirience with S60 application. One of the case of such improvement is using pointer events for creating popup text editing menu **on the event** of some text were selected with stylus in text editor control:



Preconditions

Container control wich holds text editor control should be implemented, e.g. class derived from CCoeControl and contains CEikEdwin. To handle pointer events CCoeControl::HandlePointerEventL() should be reimplemented:

```
class CTestTouchUiAppView : public CCoeControl, public MEikMenuObserver, public
MAknLongTapDetectorCallBack
{

...
private:
        virtual void HandlePointerEventL(const TPointerEvent& aPointerEvent);

        // from MAknLongTapDetectorCallBack
        void HandleLongTapEventL( const TPoint& aPenEventLocation,
                                  const TPoint& aPenEventScreenLocation );

        // From CCoeControl
```


(C) Copyright Nokia 2014. All rights reserved.

```
        TInt CountComponentControls() const;

        CCoeControl* ComponentControl(TInt aIndex) const;

        void CreatePopupMenuL( const TPoint &aPosition);

        // from MEikMenuObserver
        void ProcessCommandL(TInt aCommandId);
...
private:
 // Member variables
 CEikEdwin* iEditor;

 CAknStylusPopUpMenu* iPopupMenu;

 // Indicates whether popup menu could be displayed on TPointerEvent::EButton1Up event
        TBool iShowPopup;

        CAknLongTapDetector* iLongTapDetector;

};
```

MMP file

The following libraries are required:

```
LIBRARY cone.lib
```

```
LIBRARY eikcore.lib
```

```
LIBRARY avkon.lib
```

Source file

```
void CTestTouchUiAppView::HandlePointerEventL(const TPointerEvent& aPointerEvent)
{
 if ( AknLayoutUtils::PenEnabled() )
 {
  // Call base class HandlePointerEventL()
  CCoeControl::HandlePointerEventL(aPointerEvent);

  if (aPointerEvent.iType == TPointerEvent::EDrag)
  {
   iShowPopup = ETrue;
  }
  else if (aPointerEvent.iType == TPointerEvent::EButton1Up)
  {
   if (iShowPopup)
   {
```


(C) Copyright Nokia 2014. All rights reserved.

```
    if (iEditor->SelectionLength() > 0)
     CreatePopupMenuL(aPointerEvent.iPosition);
   }

   iShowPopup = EFalse;
  }

     iLongTapDetector->PointerEventL( aEvent );
 }

}
```

Here we use boolean flag iShowPopup, which is set to ETrue if any drag event occures. When TPointerEvent::EButton1Up event occures we check whether some text has been selected and if so - we show text edit popup menu. Otherwise ignore this event.

And for the case of long tap event we show if necessary popup menu with 'Paste' option (i.e. check whether appropriate data available from clipboard):

```
void CTestTouchUiAppView::HandleLongTapEventL( const TPoint& aPenEventLocation,
                                     const TPoint& aPenEventScreenLocation )
{
        // Here should go check of the clipboard for available data
 CreatePopupMenuL(aPenEventLocation);
}
```

The popup menu creation method itself:

```
void CTestTouchUiAppView::CreatePopupMenuL( const TPoint &aPosition )
{
 delete iPopupMenu;

 iPopupMenu = CAknStylusPopUpMenu::NewL( this, aPosition );

 if (iShowPopupMenu)
 {
 iPopupMenu->AddMenuItemL( KMenuItemCopy, EMenuItemCopy );
 iPopupMenu->AddMenuItemL( KMenuItemCut, EMenuItemCut );
 iPopupMenu->AddMenuItemL( KMenuItemPaste, EMenuItemPaste );
 iPopupMenu->AddMenuItemL( KMenuItemChangeFont, EMenuItemChangeFont );
 }
 else
 {
 iPopupMenu->AddMenuItemL(KMenuItemPaste, EMenuItemPaste);
 }

 iPopupMenu->ShowMenu();
}
```

Postconditions

Text editing options are enriched for the case of touch ui.

(C) Copyright Nokia 2014. All rights reserved.

See also

Working with Stylus Pop-up Menu API - S60 Touch UI

Working with LongTapDetector API - S60 Touch UI

Example source code

Media:Popup edit menu.zip

(C) Copyright Nokia 2014. All rights reserved.