# Archived:ScrollPane component in FlashLite

Archived: This article is **archived** because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

We do not recommend Flash Lite development on current Nokia devices, and all Flash Lite articles on this wiki have been archived. Flash Lite has been removed from all Nokia Asha and recent Series 40 devices and has limited support on Symbian. Specific information for Nokia Belle is available in Flash Lite on Nokia Browser for Symbian. Specific information for OLD Series 40 and Symbian devices is available in the Flash Lite Developers Library.

## Introduction to the ScrollPane Component

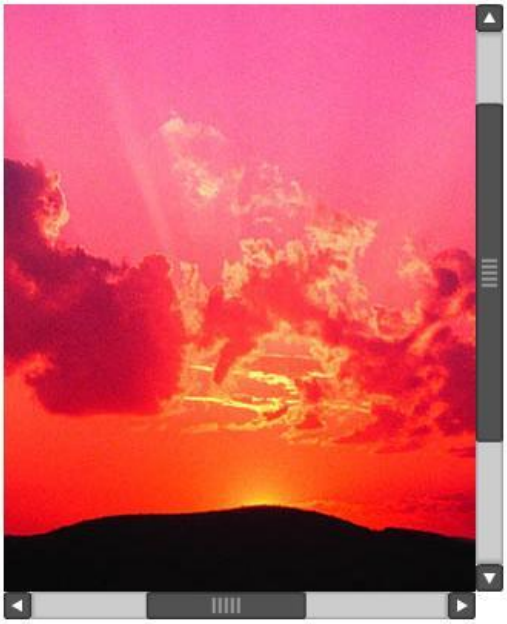
The Flash Lite ScrollPane component is a flexible and scalable user interface component that allows Flash Lite developers to create mobile user interfaces easily, using Flash Lite. The ScrollPane component uses placeholder MovieClips similar to Nokia Developer UI components in order to ease scaling, orientation, and positioning issues.

The same component can be used with equal ease for both, **touch based** and **non-touche based** mobile devices and offers all the features of desktop version of Flash CS 3 ScrollPane component. In addition, this ScrollPane component can accommodate for **dynamic change in the source content** and adjust the scrolls accordingly.

This ScrollPane component can be used for applications developed using **FlashLite 2.x** and **FlashLite 3.x**. This ScrollPane component is tested with FlashLite 2.x and FlashLite 3.x content for following

Nokia devices.

- Non touch based Nokia E-63.
- Touch based Nokia 5800 XpressMusic.
- Touch based Nokia 5530 XpressMusic.



## Requirements

- Adobe Flash Professional CS3 or CS4.
- Flash Lite 2.0 Player and above.

## Download

ScrollPane component can be downloaded here: Media:ScrollPaneComponent.zip.

## Installation

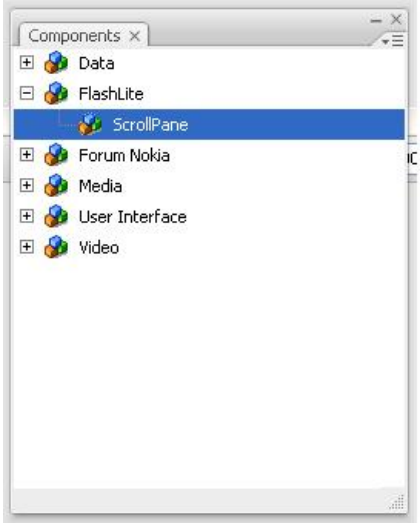

(C) Copyright Nokia 2014. All rights reserved.

Installation of the ScrollPane component is easy. Execute the components MXP file and follow the simple instructions in Adobe Extension Manager to complete the installation process. Restart Adobe Flash CS3 after installation.

Note: Component FLA and AS files should appear in the directories [INSTALL PATH]\Flash CS3\en\Configuration\Components\ScrollPane and [INSTALL PATH]\Flash CS3\en\Configuration\Classes\com\ScrollPane. In some setup environments, the files may appear under the wrong language directories (for example, fi instead of en). In such cases, the files should be manually moved to the correct directories as specified above.

## Preparations

1. Create a Flash Lite Project. Open the Component panel (Ctrl/Apple key + F7) and drag the ScrollPane component onto the stage. Assign a unique instance name for the object using the properties panel (Ctrl/Apple key + F3).




2. It is recommended to add the following lines of ActionScript code to the first frame of the project so that it can be used with some other Nokia Developer components.

```
fscommand2("DisableKeypadCompatibilityMode");
fscommand2("FullScreen", true);
fscommand2("SetQuality", "high");
Stage.scaleMode = "noScale";
Stage.align = "TL";
_focusrect = false;
```

3. In order for ScrollPane component to work properly, the **Registration point** of source movieclip must be **top-left**. Also X and Y coordinates of the ScrollPane component should be same as that of source movieclip.

4. The ScrollPane component dispatches an event when the slider is moved to make the source content scroll. This is done by calling addEventListener.

```
var myListener:Object = new Object ();
ScrollPaneInstance.addEventListener ("scroll",myListener);
myListener.scroll = function (oEvent:Object):Void
{
 //You can get the Vertical Scroll Position by oEvent.target.vPosition;
 trace ("vPosition::" + oEvent.target.vPosition);
 //You can get the Horizontal Scroll Position by oEvent.target.hPosition;
 trace ("hPosition::" + oEvent.target.hPosition);
};
```

5. The same component works for both the touch and non-touch based devices and if you want to make the ScrollPane react to device keyboard and scroll content accordingly, you need to add the keyboard listener by simply calling the following function. There are three combinations you can use the component with:


(C) Copyright Nokia 2014. All rights reserved.

- Devices with only keyboard support(Non-touch based)(Eg. Nokia E52 Nokia 3230)
- Devices with only Touch interaction. (Eg. XpressMusic 5800)
- Devices supporting both Touch and Keyboard interaction. (Eg. N97)

```
mcScrollPane.setScrollUsingKey (false);
```

Note: For devices using only touch based interaction(5800,5530), you should not call this function.

The boolean parameter will tell if the touch-based interaction is supported in addition to the keyboard support.

```
//For Nokia E-63, E-52 alike
mcScrollPane.setScrollUsingKey (false);
//For Nokia E-97
mcScrollPane.setScrollUsingKey (true);
```

The component offers the flexibility of scrolling the content using any key you may find suitable for your application. Also the keycode for keys is different in case of different devices. So you should specify the Keycodes of keys(which should be mapped for Scroll up,down,left,right)for this particular device using the Component Inspector/calling **setKeyCodes** API.

6. The ScrollPane component can accommodate the dynamic change in the source content and adjust the Scrollbars accordingly. If you have changed the source content and want to reflect that in the ScrollPane, you need to call the following function.

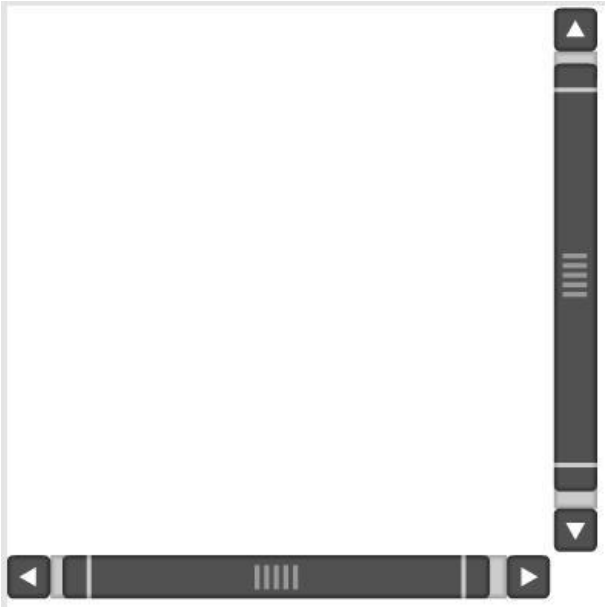
```
ScrollPaneInstance.refreshPane();
```

Note: Please note that this ScrollPane is attached to same Source instance available on the stage and hence you do not need to call the **refreshPane API**, if the Source content is not changed in size. Any other modifications to the source content will be automatically reflected.

# Structure

## Skins

The skin has separate Movieclips for Up, Over and Down states of the ScrollPane. The skin consists of **Active** and **Inactive** states of the Arrow buttons and background, top, body, and bottom part of the sliders. This structure enables the slider interface to be completely scalable and easily skinnable.



Skins have separate MovieClips for Up and Down states of the slider. The structure of these parts is defined below:

_skins


(C) Copyright Nokia 2014. All rights reserved.

- Default
  - ActiveState
    - Down
      - Arrow_bottom_dn
      - Arrow_left_dn
      - Arrow_right_dn
      - Arrow_top_dn
      - Scroll_horizontal_bg_dn
      - Scroll_vertical_bg_dn
      - Slider_horizontal_body_dn
      - Slider_horizontal_left_dn
      - Slider_horizontal_lines_dn
      - Slider_horizontal_right_dn
      - Slider_vertical_body_dn
      - Slider_vertical_bottom_dn
      - Slider_vertical_lines_dn
      - Slider_vertical_top_dn
    - Over
      - Arrow_bottom_or
      - Arrow_left_or
      - Arrow_right_or
      - Arrow_top_or
      - Scroll_horizontal_bg_or
      - Scroll_vertical_bg_or
      - Slider_horizontal_body_or
      - Slider_horizontal_left_or
      - Slider_horizontal_lines_or
      - Slider_horizontal_right_or
      - Slider_vertical_body_or
      - Slider_vertical_bottom_or
      - Slider_vertical_lines_or
      - Slider_vertical_top_or
    - Up
      - Arrow_bottom_up
      - Arrow_left_up
      - Arrow_right_up
      - Arrow_top_up
      - Scroll_horizontal_bg_up
      - Scroll_vertical_bg_up
      - Slider_horizontal_body_up
      - Slider_horizontal_left_up
      - Slider_horizontal_lines_up
      - Slider_horizontal_right_up
      - Slider_vertical_body_up
      - Slider_vertical_bottom_up
      - Slider_vertical_lines_up
      - Slider_vertical_top_up
  - InactiveState
    - Arrow_bottom_up
    - Arrow_left_up
    - Arrow_right_up
    - Arrow_top_up


(C) Copyright Nokia 2014. All rights reserved.

Note: Each new custom skin must follow the same folder structure, with linkage identifiers named accordingly.

## Inspectable Parameters

Inspectable parameters help customise the ScrollPane component from the Component Inspector panel (Shift + F7). All inspectable parameters can also be controlled via ActionScript with the component APIs.

| Parameter | Description | Value |
|---|---|---|
| AssetsSkins | Defines the component's skin linkage identifier path. For instance, for the 'mySkin' value, it would search for the skin components under the ScrollPane.mySkin.* linkage path. | Skin identifier string |
| Height | Height of the ScrollPane | Number (Default is 100) |
| hLineScrollSize | The number of pixels to move the content when an arrow in the horizontal scroll bar is clicked. | Number (Default is 10) |
| Target | A string that indicates the Source content movieclip to which the ScrollPane will be added. | Instance of Source MovieClip |
| vLineScrollSize | The number of pixels to move the content when an arrow in the vertical scroll bar is clicked. | Number (Default is 10) |
| Width | Width of the ScrollPane | Number (Default is 160) |
| KeyCodes | The array with Keycode of keys in the sequence:Scroll Up, Scroll Left, Scroll Right, Scroll Down | Array (Default is [50,52,54,56]) |

# ScrollPane Component ActionScript API

## Setting the skin path

*public function setAssetsSkin (path:String):Void*

Sets a new path for the skin. Reconstructs the component

## Enable/Disable the component

*public function enableDisableComponent (bState:Boolean):Void*

Enable and disable the component based upon the Boolean flag parameter. Setting the flag to false makes the component disable and stop reacting to user input. Setting the flag to true makes the component enable and react to user input.

## Changing ScrollPane Dimension

*public function changeScrollPaneDimensions (nWidth:Number, nHeight:Number, bMaintainXY:Boolean):Void*


(C) Copyright Nokia 2014. All rights reserved.

Sets the new height and width for the ScrollPane component.

If you are using the component with devices supporting change in orientation, you can add an orientation listener and change the dimension of ScrollPane to suit to the current orientation. The last parameter (bMaintainXY) will be useful in case you want to maintain the same state of the source content area (or the visible area at the top) within the ScrollPane even after the change in dimension. Setting the parameter to true will change the dimension of the ScrollPane and also restore the source content at the same state. That is, if before changing the dimension the ScrollPane showed a input text-field at the top, then even after changing the dimension, the same text-field will be visible at the same offset within the ScrollPane.

## Setting ScrollPane Location

*public function setScrollPaneXY (nX:Number, nY:Number):Void*

Sets the X and Y co-ordinates of the ScrollPane.

## Setting the ScrollPane Visibility

*public function setVisibility (bVisibility:Boolean):Void*

Sets the ScrollPane visibility according to the Boolean flag. If set to false makes the ScrollPane component invisible and setting to true makes it visible.

## Setting Vertical Position of ScrollPane content

*public function setVPosition (nY:Number):Void*

Orients the scroll pane's contents in pixels, and adjusts the vertical scroll box (thumb) proportionally. The 0 position is at the top end of the scroll track, which causes the top edge of the scroll pane content to be visible in the scroll pane.

## Setting Horizontal Position of ScrollPane content

*public function setHPosition (nX:Number):Void*

Orients the scroll pane's contents in pixels, and adjusts the horizontal scroll box (thumb) proportionally. The 0 position is at the left end of the scroll track, which causes the left edge of the scroll pane content to be visible in the scroll pane.

## Setting Vertical Line Scroll Size

*public function setVLineScrollSize (nPixels:Number):Void*

Sets the number of pixels to move the content in the display area when the user clicks a scroll arrow in a vertical scroll bar. The default value is 10.

## Setting Horizontal Line Scroll Size

*public function setHLineScrollSize (nPixels:Number):Void*

Sets the number of pixels to move the content when an arrow in the horizontal scroll bar is clicked. The default value is 10.

## Refresh ScrollPane for the changed Source content

*public function refreshPane ():Void*

If the source content is changed in the dimension you can call the refresh API. This will make sure that the ScrollPane will work with the modified source content by resetting the scroll bar. There is no need to refresh the ScrollPane in case the source content is same in dimension and merely changes some input property (for example, a text field).

## Add Keyboard Scrolling Support

*public function setScrollUsingKey (bFlag:Boolean):Void*

Sets the Keyboard input support for the ScrollPane component.

As mentioned earlier, this ScrollPane component works with Touch based and non touch based devices. The component works by default for only Touch based devices. But if you want to add the Keyboard interaction, you should call this API. If the target device supports only Keyboard interaction, you should call the API with Boolean flag as 'false' and if the target device supports both touch based and keyboard interaction(and you wish to add both support for your application), you should call the API with Boolean flag as 'true'.


(C) Copyright Nokia 2014. All rights reserved.

### Setting Keycode of ScrollKeys

*public function setKeyCodes (aKeyCodes:Array):Void*

Sets the Keycode of the keys mapped for Scroll Up, Left, Right, Down functionality.

## Demo Examples

The attached Examples will show how to use the ScrollPane component in FlashLite Applications.

- One is for touch based Nokia 5800 XpressMusic Device.
- Other is for non-touch based and keyboard supported Nokia E-63. Can scroll using 2,4,6,8 numbered keys in addition to ScrollPane Left/Right/Top/Bottom buttons.


(C) Copyright Nokia 2014. All rights reserved.