

Q: It seems like the `scrollView` ignores the `frame.origin` of its `docView`. Also, does a `ScrollView` clip to its update `rect(s)`?

A: A `ScrollView` translates the coordinate system of its `contentView` in response to user movement of the scrollers. Since the frame of the `docView` is defined in the coordinate system of the `contentView` (i.e. it's a subview of the `contentView`), translating the coordinate system of the `contentView` has the effect of moving the frame, and hence the visible portion of the `docView`.

When you do a **`setDocView:`** the system translates the coordinate system of the `contentView` so that the `contentView`'s `bounds.origin` is the same as the `frame.origin` of the `docView`. For example: if the `frame.origin` of your `docView` is `{100.0,100.0}`, the `bounds.origin` of the `contentView` will be `{100.0,100.0}` as well.

So the `ScrollView` does pay attention to what the `frame.origin` of the `docView` is, but it doesn't really matter what it is, at least initially.

User code really shouldn't make changes to the frame of the `docView` as it is being managed by its superview (a `ClipView`). To achieve scrolling use the - **`rawScroll:`** method.

No, the drawing is not clipped to the update rects; however, you can restrict the area being redrawn yourself by using **`PSrectclip()`** or **`NXRectClip()`** in your **`drawSelf::`** method.

QA160

Valid for 1.0, 2.0, 3.0

Q: In my application I have subclassed the Text object and called it SubText. I have a ScrollView which I installed using InterfaceBuilder which has a standard Text object as its docView. I wish to replace that Text object with my own but I wish to maintain all the same behaviors with a vertical scrollbar and text which resizes properly. How do I do that?

A: The following code snippet creates an instance of your new subclass of Text and inserts it into the docView of the ScrollView. The old Text object is then freed. In the code sample ``theSV" is an outlet connected to the ScrollView. Because an existing ScrollView is used, certain setups are not required. See Chapter 9, page 35 of the Concepts volume of the 1.0 Technical Documentation for more information on setting up a brand-new ScrollView.

The key here that many people miss is to set the min size and max size on the Text object.

```
id      stdDoc;  
NXRect r;  
  
/* Measure the old doc view. */  
stdDoc = [theSV docView];  
[stdDoc setFrame:&r];
```

```
[theSV setVertScrollerRequired:YES];  
[theSV setHorizScrollerRequired:NO];  
[theSV setDynamicScrolling:YES];
```

```
/* Instantiate the new Text object */
```

```
#ifdef 1.0
```

```
newDoc = [SubText newFrame:&r];
```

```
#else /* 2.0 or 3.0 */
```

```
newDoc = [[SubText alloc] initWithFrame: &r];
```

```
#endif
```

```
[newDoc moveTo:0.0:0.0];
```

```
[newDoc notifyAncestorWhenFrameChanged: YES];
```

```
[newDoc setVertResizable:YES];
```

```
[newDoc setSelectable:YES];
```

```
[newDoc setEditable: YES];
```

```
[newDoc setAutosizing:NX_WIDTHSIZABLE];
```

```
[newDoc setMinSize:&r.size];
```

```
r.size.height = 1.0e30;
```

```
[newDoc setMaxSize:&r.size];
```

```
[theSV setDocView:newDoc];
```

```
/* Free the old Text object */
```

```
[stdDoc free];
```

Note that the ifdef for 1.0 is for illustration purposes only and is NOT defined in any NeXTstep include file.

QA555

Valid for 1.0, 2.0, 3.0