

Displaying STATUSLINE Messages

OpenInsight provides a convenient property named STATUSLINE as a way to direct help text for menu items and bitmap buttons to a particular edit line control. Often this edit line sits at the bottom of an MDI Frame or other primary window where most users are trained to look for various application messages. For most status bar uses, the SRP StatusBar control nicely replaces the normal edit line control, including the ability to display help text for SRP Button controls (viz. *SRP Status* option in the [PaneType](#) property). This article explains how the SRP StatusBar can be used to display help text for regular OpenInsight menu items and bitmap buttons.

Getting By With a Little Help From Our Friends

The nature of the STATUSLINE property cannot be changed on our end. Therefore, we must rely on a persistent edit line that has been set up to be the STATUSLINE property for a given window (n.b. the STATUSLINE property must be set for each OpenInsight form but the same edit line control can be used for any or all forms.) This gives us a way to get the help text. This edit line should also be made invisible so that users never see it.

Then we just need to add some event handler logic in the edit line's CHANGED event, for example:

```
Set_Property(@Window : ".OLE_FRAME_STATUS", "OLE.PaneCaption[1]", Param1)
```

Voilà, by capturing the help text messages within the CHANGED event handler of the STATUSLINE control and then sending this to our [PaneCaption](#) property, the SRP StatusBar control will now receive and display ordinary OpenInsight help text messages.

Beware the System Monitor

There is a rather unforgiving side effect when the form is running and the System Monitor is launched. The message chatter and event processing that is created in this situation causes the Engine to remain perpetually busy, effectively locking up OpenInsight. This would not ordinarily be a problem for regular end users, but developers need a little love so we recommend modifying the CHANGED event handler logic to include a few more lines of code:

```
MonitorHandle = FindWindow('RTI_PSMonitor' : \00\, '')
MonitorVisible = IsWindowVisible(MonitorHandle)
If MonitorVisible then Set_Property(@Window, 'STATUSLINE', '')
Set_Property(@Window : '.OLE_FRAME_STATUS', 'OLE.PaneCaption[1]', Param1)
```

The goal here is to first see if the System Monitor is visible by using the FindWindow and IsWindowVisible Windows API calls. If so then turn off the STATUSLINE property. Obviously you can adapt this logic in other areas of your application to automatically turn the STATUSLINE property back on.