Registering Events

In order to handle SRP ActiveX Control events in OpenInsight, you need to make a request to OI–in code–that you want that event passed to you. To do this, you send the control a QUALIFY_EVENT message.

ALL_OLES

When sending the qualify event message, instead of passing the name of a specific event, you can pass "ALL_OLES". While convenient, we actually do not recommend this. Processing events has overhead, and if there are events that fire quite frequently, such as every time the mouse moves, you will incur noticeable lag as OI attempts to process every tiny movement of the mouse. This could lead to dozens of events fired every second. Even if you don't handle the event, OI still has to convert the event's parameters and pass it through the event chain. Since OpenInsight is single-threaded, CPU time spent sending porcessing events you don't need is CPU time that could be better spent elsewhere.

Synchronous or Asynchronous

When you qualify an event, you can pass additional fields. Field <4> specifies one of three values that tell OpenInsight how to process the event:

Value	Behavior
0	Asynchronous. This is the default behavior. When the event occurs, it is placed in the event queue to be processed as soon as the ActiveX control is done processing.
1	Synchronous. When the event occurs, it is pushed through the event chain immediately, temporarily interrupting the ActiveX control. When the event is done processing, the ActiveX control is allows to finish processing.
2	Recursive Synchronous. Same as the above, but OI will also immediately process other ActiveX events that fire as a result of calling other properties and methods of the control.

Synchronous behavior was added to allow programmers to process events that needed immediate feedback. For example, the SRP EditTable Control's BeforeUpdate event allows you to set the Cancel property which will prevent the cell from updating, thus offering a convenient way to handle validation.

Since values 1 and 2 cause events to be handled immediately, you might be tempted to qualify all your events synchronously. This, however, will not lead to better performance and can often lead to problems. Keep in mind that OI is single-threaded, and ActiveX controls run on that thread. Let's say the user clicks on an SRP Button Control. The button takes over OI's only thread and does something like this:

1. Validate the button was clicked
2. Update the button state
3. Fire the OnClick event
4. Redraw the button

If OnClick was qualified asynchronously (0), then the overall process will look like this:

1. Validate the button was clicked
2. Update the button state
3. Fire the OnClick event
4. Redraw the button
5. OI processes the OnClick event

If, however, it was qualified synchronously (1 or 2), the order looks like this:

1. Validate the button was clicked
2. Update the button state
3. Fire the OnClick event
4. OI processes the OnClick event
5. Redraw the button

Surprisingly, the second scenario will appear less performant to the end user. When the event gets processed, that will take time, and the button will wait to draw its new state until OI is done processing it. It's better to think of OLE events as polite notifications rather than demands for immediate processing. There are a few exceptions listed below, but as a rule of thumb, qualify your events asynchronously.

Focus Events

While qualifying most events synchronously is harmless, events such as OnGotFocus or OnLostFocus must be asynchronous. These events occur when the control has received focus, which prompts the control to perform a lot updates to its state, but OI also wants to do a lot of stuff when focus changes. If these events are synchronous, OI will try to update the state of other controls and the form while the ActiveX control was in the middle of updating it's own state. This leads to focus flicker and unwanted clicks.

Always qualify these two events asynchronously.

Events to Qualify Synchronously

These are the only SRP ActiveX Control events that demand immediate feedback and therefore need to be synchronous (i.e., set Qualifier<4> to 2):

SRP EditTable

- BeforeDeleteRecords
- BeforeInsertRecords
- BeforeNewRecords
- BeforeUpdate
- OnInvalidData
- PosChanging

SRP Schedule

- BeforeApptDrag
- BeforeApptDrop
- BeforeNewApptDrag
- BeforeNewApptDrop

SRP Tree

- BeforeUpdate
- OnDragStart

Dialog Boxes

In OI 9.x or earlier, synchronous events do not work in modal dialog boxes. All events must be qualified asynchronously.