





## AlarmDisplayList

AlarmDisplayList			
AlarmDisplayList1			
Lse	Server0		
0	0		
MaxNo	Refresh		
100	0		
ColumnChoice	Acknowledge		
0	0		
coAlarmTitle	Delete		
0	0		
coSystemLogging	NoActive		
0	0		
ccAlarmChanged	HelpTxt		
0	0		
	AlaTxtShort		
	0		
	AlaTxtLong		
	0		

Diese Klasse dient dem Aufbau der aktuellen Alarm-Liste in einem LASAL Screen-Projekt. Darüber hinaus werden die beiden Alarm-Texte Short und Long für die Alarm-Details zur Verfügung gestellt (auch Alarm Beschreibung genannt).

### Ausgabe der Alarmliste in der Visualisierung

Damit die Klasse auf die Kernelfunktionen zugreifen kann, muss zuerst eine Verbindung des Clients Lse mit einem \_Lse-Objekt erfolgen. Um nun die Alarmliste darzustellen, ist der Server0 als NumEdit im LSE Projekt zu platzieren. Als Virtual Objectname vergibt man im LSE den Objektnamen der platzierten Klasse (z.B. AlarmDisplayList1).

Außerdem kann mithilfe des Clients coSystemLogging, welcher mit einem SystemLogging-Objekt verbunden werden muss, mitgeloggt werden.

## Spaltenanzeige definieren

Es können die Spalten definiert werden, welche in der Visualisierung angezeigt werden sollen. Der Client ColumnChoice kann folgendermaßen gesetzt werden, wobei 1 = aktiv:

- 2#1000 = Alarmnummer
- 2#0100 = Zeit gekommen/gegangen
- 2#0010 = Zyklusnummer ("Para1")
- 2#0001 = Alarmtext

## Einstellen der Ringpuffergröße

Mit dem Client MaxNo wird die maximale Anzahl an Einträgen im Ringpuffer definiert. Als Eintrag gilt ein Alarm.

## Schnittstellen

### Clients

<b>Lse</b>	Objekt-Kanal zum LSE-Objekt Datentyp	Objektkanal zur Klasse _Lse
<b>MaxNo</b>	maximale Anzahl an Einträgen im Ringpuffer Datentyp	UDINT
<b>ColumnChoice</b>	Bitmuster zum Enablen der Darstellung von Display-Items: 2#1000 = Alarmnummer 2#0100 = Zeit gekommen/gegangen 2#0010 = Zyklusnummer (KaiAnd: real "Para1") 2#0001 = Alarmtext Datentyp	BDINT
<b>coAlarmTitle</b>	Objekt-Kanal zu AlarmTitleLine zum Beeinflussen des Listen-Headers (optional) Datentyp	Objektkanal zur Klasse AlarmTitleLine
<b>coSystemLogging</b>	Objekt-Kanal zu Logging-Funktion (optional) Datentyp	Objektkanal zur Klasse AlarmTitleLine
<b>ccAlarmChanged</b>	Kommandokanal um Alarmänderungen zu teilen (optional) Datentyp	DINT

## Server

<b>Server0</b>	(Server0 geerbt von _myIO)			
	Einheit	-	Datentyp	DINT
	Wertebereich	-	Write Protected	FALSE
	Defaultwert	-	Retentive	FALSE
<b>Refresh</b>	(Refresh geerbt von _myIO)			
	Einheit	-	Datentyp	DINT
	Wertebereich	-	Write Protected	FALSE
	Defaultwert	-	Retentive	FALSE
<b>Acknowledge</b>	-1       quittiere alle Alarme +x       quittiere ausgewählten Alarm			
	Einheit	-	Datentyp	DINT
	Wertebereich	-1 ... +x	Write Protected	FALSE
	Defaultwert	-	Retentive	FALSE
<b>Delete</b>	-1       lösche alle Alarme +x       lösche ausgewählten Alarm			
	Einheit	-	Datentyp	DINT
	Wertebereich	-1 ... +x	Write Protected	FALSE
	Defaultwert	-	Retentive	FALSE
<b>NoActive</b>	Anzahl aktiver Alarme ... nur nach .Read()			
	Einheit	-	Datentyp	UDINT
	Wertebereich	-	Write Protected	TRUE
	Defaultwert	-	Retentive	FALSE
<b>HelpTxt</b>	Befehl "lese Alarm-Texte für Alarm-Details" des ausgewählten Alarms			
	Einheit	-	Datentyp	DINT
	Wertebereich	-	Write Protected	FALSE
	Defaultwert	-	Retentive	FALSE
<b>AlaTxtShort</b>	Alarm-Text Short nach dem Lese-Befehl			
	Einheit	-	Datentyp	UDINT
	Wertebereich	-	Write Protected	FALSE
	Defaultwert	-	Retentive	FALSE
<b>AlaTxtLong</b>	Alarm-Text Long nach dem Lese-Befehl			
	Einheit	-	Datentyp	UDINT
	Wertebereich	-	Write Protected	FALSE
	Defaultwert	-	Retentive	FALSE

## Globale Methoden

Bei geerbten Methoden wurden die englischen Kommentare übernommen.

<b>GetEvent</b>	<p>Wird jedes Mal aufgerufen, wenn der Eingang aktiv ist und ein Ereignis auftritt:</p> <ul style="list-style-type: none"> <li>▶ ped pointer to _EDITOR information</li> <li>▶ pe pointer to _EVENT information</li> <li>◀ retcode für die Return-Anweisung gibt es 3 Möglichkeiten               <ul style="list-style-type: none"> <li>_IDLE: das System macht weiter wie bisher (system goes on)</li> <li>_IDIDIT: der User hat es gemacht (system is ready)</li> <li>_IFAILED: der User hat es probiert, ist aber gescheitert (system is ready)</li> </ul> </li> </ul>
<b>IF_Start</b>	<p>Die Methode wird einmal direkt vor dem Zeichnen des Objekts aufgerufen, zum Beispiel bei geöffnetem Bildschirm.</p> <ul style="list-style-type: none"> <li>▶ pio Pointer auf _IO information</li> <li>▶ firsttime TRUE: wenn das System die Zeichnung nach geöffnetem Bildschirm sehen möchte FALSE: wenn das System den Hintergrund neu zeichnen möchte</li> </ul>
<b>IF_Run</b>	<p>Die Methode wird zyklisch aufgerufen, während das Objekt am Bildschirm ist</p> <ul style="list-style-type: none"> <li>▶ pio pointer to _IO information</li> <li>▶ input TRUE: es ist ein Eingang FALSE: es ist ein Ausgang</li> </ul>
<b>Line</b>	<p>Die Methode wird jedes Mal aufgerufen, wenn ein Neuzeichnen einer einzelnen Linie notwendig ist.</p> <ul style="list-style-type: none"> <li>▶ ps Pointer auf structure _SCROLL</li> <li>▶ pr Pointer auf die Position, wo die Linie gezeichnet werden soll</li> <li>▶ line Nummer der Linie, die gezeichnet werden soll</li> <li>▶ state TRUE: Linie wird gewählt FALSE: passiv</li> </ul>
<b>LineHeight</b>	<p>Gibt als retcode den zweifachen Wert von preselect aus</p> <ul style="list-style-type: none"> <li>▶ preselect Vorwahl</li> <li>◀ retcode zweifachen Wert von preselect</li> </ul>
<b>SetAlarmInfo</b>	<p>Liest zwei Alarm-Texte aus und schreibt diese auf Objekte "AlarmTxtShort" und "AlarmTxtLong" /für die Visualisierung)</p> <ul style="list-style-type: none"> <li>▶ psa Pointer auf ein SingleAlarm-Objekt</li> </ul>