Base Language - Feature #12012

raise 4GL warnings as warnings, so SUPPRESS-WARNINGS applies to them

10/15/2026 09:31 PM - Greg Shah

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**reviewer:**	
vendor_id:	GCD	**production:**	No
case_num:		**env_name:**	
version_reported:		**topics:**	
version_resolved:			

Description

History

#1 - 10/15/2026 09:31 PM - Greg Shah

Raise 4GL warnings as warnings, so SUPPRESS-WARNINGS applies to them

Project: Base Language
Related: #5901 (some 4GL warnings must process differently when the current scope has a CATCH block)

Problem

SESSION:SUPPRESS-WARNINGS and SUPPRESS-WARNINGS-LIST are implemented in ErrorManager (isSuppressWarnings, around lines 1453, 1484, 3442 and 3791). They suppress whatever FWD raises through the warning path (recordOrShowWarning and the warning mode of recordOrShowError).

The problem is classification. Some conditions that the 4GL raises as a **warning** are raised by FWD as an **error**, through recordOrThrowError, so the suppression never sees them. An application commonly sets SUPPRESS-WARNINGS = TRUE once at startup and leaves it on. Under FWD, each misclassified condition then appears to the user as an error message, or raises the ERROR condition and changes the flow of control, where OpenEdge stays silent and continues.

There is also a defect in setSuppressWarningsList (near ErrorManager.java:3823). A successful assignment adds the new numbers to the existing set without clearing it first, so a second assignment does not replace the first. Only an invalid entry clears the set.

Work

1. Build an inventory of the messages FWD raises with recordOrThrowError or displayError. For each, establish from OpenEdge whether the 4GL issues it as an error or as a warning. Its documentation, PROMSGS and testcases are the sources. There are roughly 500 recordOrThrowError calls in util alone, so start with the conditions most likely to occur in ordinary programs:
   - numeric overflow and display format (for example the "value does not fit" family);
   - string length;
   - -mmax / -s type resource warnings;
   - conversions and widget attribute assignments.
2. Route each warning-class condition through the warning path, so that:
   - SUPPRESS-WARNINGS suppresses its display;
   - SUPPRESS-WARNINGS-LIST suppresses it by number;
   - it does not raise ERROR and does not set ERROR-STATUS:ERROR, but is recorded in ERROR-STATUS the way the 4GL records it, including under NO-ERROR.
3. Make SUPPRESS-WARNINGS-LIST replace the previous list on assignment.
4. Check that suppression is honoured wherever the message is displayed, on the server and on the client.
5. Coordinate with #5901, which covers how warnings behave inside a scope with a CATCH block.

Testcases

For each reclassified message, a testcase run against OpenEdge for the baseline that covers:

- the message with SUPPRESS-WARNINGS off and on;
- the message with its number in, and not in, SUPPRESS-WARNINGS-LIST;
- the message with and without NO-ERROR, checking ERROR-STATUS:ERROR, NUM-MESSAGES and GET-NUMBER;
- that flow of control continues after the warning.

Also add a testcase for replacing SUPPRESS-WARNINGS-LIST.