User Interface - Feature #12048

replacement for the DBI ctGrid ActiveX control (editable grid)

10/20/2026 05:51 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 - 10/20/2026 05:51 PM - Greg Shah

Replacement for the DBI ctGrid ActiveX control (editable grid)

Project: User Interface

Problem

ctGrid (ctGrid.ocx 2.0.9) is an editable grid with combo-box cells, populated from temp-tables. Members used include AddComboItem (by far the most used), CellText, AddColumn, ItemCargo, EditCell, EditColumn, EditRecord, CommitEdit, CancelEdit, SelectOnEdit, CellStyle, CellTime and ColumnSortable. Events: Change, BeforeCellEdit, ComboItemSelected, RightClick, CellKeyDown, CheckClick, AfterCellExit.

Two patterns are typical: line-item editors with cell-level validation, and a mapping grid paired with a preview grid for flat-file imports. Several grids may sit on the tabs of a tab control.

FWD's TREELIST has no cell editors or cell-level edit events, and the native editable BROWSE has a different model.

Work

1. Write a replacement control with the ctGrid model: editable cells of text, combo, check and time types, the edit life cycle (EditCell, CommitEdit, CancelEdit) and the cell-level events with the vendor's ordering and veto semantics.
2. Share the cell editors with the ctList list grid where that is natural.
3. Register the control's CLSID in the OCX registry (ocx-registry.xml), mapping it to the target widget, as #11919 did for ctToolBar.
4. Decode the design-time state the control keeps in the .wrx storage at conversion time, so that it exists after conversion.
5. Implement the members and events listed above on the target, with the vendor's semantics: argument types, return values, defaults, and the order in which events fire. Members of the type library that are not used become no-ops with a conversion warning, as the OCX pipeline already does.
6. Follow the OCX emulation playbook attached to #11919: an OpenEdge test program with goldens for each member and event, recorded input sequences for the interactive parts, and Lyra automation.
7. Update the gap marking and the OCX registry documentation.

Testcases

Each member and event above, exercised on Windows OpenEdge with the real control to produce goldens, then run against the FWD replacement on the web and Swing clients. The vendor help files are the reference where behaviour cannot be observed directly.