User Interface - Feature #12057

replacement for the ImageKnife (Picbuf) ActiveX control

10/20/2026 06:14 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 06:14 PM - Greg Shah

Replacement for the ImageKnife (Picbuf) ActiveX control

Project: User Interface

Problem

ImageKnife/OCX (oik32.ocx, Media Architects, later Olympus; long abandoned) is an image buffer control. Applications use it for banks of photo thumbnails with a main viewer: Picbuf copies from a clicked thumbnail into the viewer, FileName and Load, Picture, AutoScale, AutoSize, ScaleImage, ColorDepth, OafFormat, Store, Assign, Clear. Events: Click, MouseMove and DblClick. It is also used to resize and merge images into thumbnails for an icon store.

FWD has the native IMAGE widget, with stretch-to-fit, retain-shape and mouse events, but no image buffer model and no image processing.

Work

1. Implement a new Picbuf widget with the control's image buffer model, its mouse events and its display options (AutoScale, AutoSize). Do the image processing — scaling, thumbnail generation, merging, colour depth and format conversion — in Java on the server.
2. Support loading from and storing to a file on the server, and buffer-to-buffer copies.
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.