WPF Control Reference

Embed WPF controls for custom functionality.

Reference **Controls** **Viewer** WPF

WPF Controls allows to browse a generic WPF control pointing directly to the DLL file. This extensible component is available to WPF display types.

To build your own control for this element, see WPF Control API Reference.

Requirements
Configuration
Runtime Execution
Related

For frequently used User Controls or those with many configuration settings, it is advisable to use the Portable API to encapsulate them with a custom Configuration Dialog. The Portable API also provides guidance on creating an HTML5 version of the control, should you choose to make it portable.

Requirements

This components runs only on Windows, on RichClient and SmartClient clients.

Configuration

The WPF Control enables selecting other graphical components from .DLL files.

1. Go to **Displays / Draw**.
2. Select **Viewer**, then **WPF Control**.
3. Select the .DLL file at your choice. Then, click **OK**.
4. Click on the Draw area to add the component.
5. Double-click the object to open the configuration window.

Each selected component has a Properties table with the following columns:

WPF Control Settings	
Field	**Description**
Name	Identifies by a unique name specified by the user.
Type	The control type is selected from a list of available .DLL files, defining the specific control and its capabilities.
Assembly	The .DLL file path provides the location of the assembly that contains the control definition, enabling its use in the application.
Change path	Allows users to modify the file path of the .DLL assembly containing the selected control. By clicking this button, users can navigate to a different directory or file location to select a new .DLL file. This feature is useful when updating to a newer version of the assembly, switching to a different control library, or correcting an incorrect file path. Changing the path ensures that the application uses the correct assembly for the control, maintaining functionality and compatibility.

Properties	Lists the configurable attributes of the selected control. Each property influences a specific aspect of the control's behavior or appearance. The columns under Properties include: **ControlProperty**: Specifies the particular property of the control that can be configured. **Binding**: Determines the data binding mode for each property, dictating how data flows between the UI and data source. Options include: **OneTime**: Binds the data once and does not update it thereafter. **OneWay**: Updates the UI when the data source changes but not vice versa. **TwoWay**: Allows reciprocal updates between the data source and the UI. **OneWayToSource**: Updates the data source when the UI changes but not vice versa. **SetOnLoad**: Specifies whether the property should be initialized during the control's load time. Enabling this ensures the property is set as soon as the control is loaded. **ObjectLink**: Allows users to enter values, quoted strings, or object properties to link external objects or values to the control property. This facilitates dynamic interactions and data binding, enabling the control to interact with other components or data sources.
Display object edges in configuration	Allows users to toggle the visibility of object edges during the configuration process. When enabled, it shows the outlines of objects, helping users to visually align and adjust components.

Runtime Execution

During runtime, the WPF Controls component enables the integration of generic WPF Controls by referencing the corresponding DLL files. When the application runs, the WPF Controls component loads and displays the specified WPF Control within the WPF display types. For frequently used User Controls or those with extensive configuration settings, users can utilize the Portable API to encapsulate these controls with a custom Configuration Dialog. Additionally, the Portable API offers guidance for creating an HTML5 version of the control.

Related

- See **WPF Client overlay limitations (HWND-hosted controls)** for the shared airspace constraint — when the embedded .DLL hosts native Win32 content (WindowsFormsHost, WebBrowser, or any HWND-based wrapper), WPF visuals cannot overlay it on the WPF Client. Pure managed WPF controls are not affected.
- See **WPF Control API Reference** to build your own control for this element: the project, the rules the Designer applies, and a downloadable template.

In this section...