Displays Multiple Popups Example

Manage multiple popup displays.

Download the solution Displays Multiple Popups Example.dbsln.

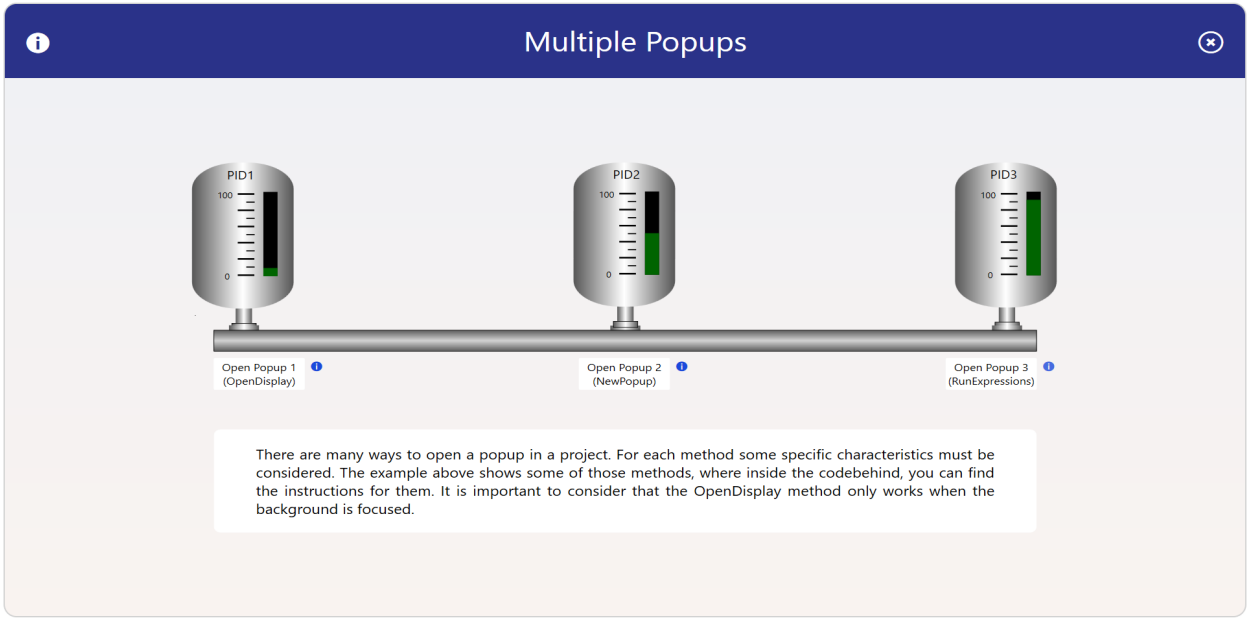
In this example we are showing different ways to open a popup in a solution.

Summary

Popups are useful tools for displaying additional information without redirecting the user to a new page. They are especially effective for showing lists of objects in a standardized way, ensuring a consistent and intuitive user experience.

In this example we are showing different ways to open a popup in a solution. The example also shows how to create reusable popup displays that can be opened side by side and mapped to different tags in the application.

Due to limitations with the OpenDisplay method, the first button will only function if the background is maximized.



Technical Information

There are three ways to open a popup in this example:

- Through the method Client.OpenDisplay() run in the Codebehind.
- Through the method Client.NewPopup() also run in the CodeBehind.
- Through the method Display.<Display Name>.NewPopup() as an expression.

Through Client.OpenDisplay():

The `OpenDisplay` method can be used to open any type of display, whether it's a standard display, a popup, a dashboard, or a dialog box. You call this method by passing the following arguments:

- Page name - as a String
- Mnemonics to be used inside the popup - also as a String separated by a semi colon.

This method is versatile, allowing you to open any display at any given time. However, it has limitations: it will not function properly if a second popup is open and focused (meaning the last click was on the popup rather than the background screen). This occurs because, when multiple displays are open simultaneously, this method prioritizes opening the display in a layer further back, excluding the focused display.

In this example, the code snippet used to run the `OpenDisplay` can be seen below.

```
@Client.OpenDisplay("PopupPID", "PID = PID1 ; NAME='PID 1'");
```

Through Client.OpenPopup():

The `OpenPopup` method works similarly to `OpenDisplay`; however, the layering order is different. Since the solution recognizes it as a popup, it will forcefully open the display in the front layer, ensuring it works as intended.

It also uses the same arguments as `OpenDisplay`, as seen below.

```
@Client.NewPopup("PopupPID", "PID = @Tag.PID2 ; NAME='PID 2'");
```

Through Display.<Display Name>.NewPopup():

As the name suggests, this method works similarly to the Client version but in a more organized way. Since we aren't passing the display name as a string, it's considered a direct reference to the display and can be found in the Cross-Reference page. This approach is beneficial for maintenance and for keeping track of what resources are being used.

This time, since the display name is already included in the method, there's no need to pass it as an argument, just the mnemonics that will be used with it. Also, this time we will run it as an expression to demonstrate the multiple ways these methods can be called.

```
Display.PopupPID.NewPopup("PID=Tag.PID3;NAME='PID 3'")
```

Open the Popup in a specific position:

You can open the Popup in a specific position using:

@Client.UI.OpenDisplay("<DisplayName>",Left,Top,Width,Height);

e.g: @Client.UI.OpenDisplay("Popup",500,200,300,300);

It will open 500px from the left, 200px from the top and a 300x300 size popup

> You can leave the Width and Height empty so it will use what you defined in the popup size

Multiple Popup Instances

The Multiple Popups feature allows users to create multiple instances of the same Display simultaneously using the `Display.NewPopup` and `Client.NewPopup` methods. Each popup operates independently, enabling unique configurations through input parameters such as labels, position, and size. The software platform manages `Display.NewPopup` automatically, while `Client.NewPopup` requires manual updates if the Display name changes. This feature facilitates dynamic, concurrent user interface interactions without closing existing displays.

Add a Display or Popup

There are many ways to open a new Display or Popup in a Solution. For each method, some specific characteristics must be considered. The list below shows some of those methods as well as a brief description of their behavior.

Display.Open

- Begins the Open Display process. The syntax is as follows:

`Display.<DisplayName>.Open()`

- Returns a flag (true or false) indicating success or fail.

E.g.: `@Display.MainPage.Open()`

Display.OpenModal

- Opens Display as Modal. The syntax is as follows:

`Display.<DisplayName>.OpenModal()`

- Not available for Mono and HTML5
- Returns a flag (true or false) indicating success or fail.

E.g.: `@Display.SelectPage.OpenModal()`

Display.NewPopup

- Opens a new one. If the Display is already open, a new instance will open. The syntax is as follows:

`Display.<DisplayName>.NewPopup(input parameters)`

- The accepted input item parameters are:

    LabelList[label1=tag1; label2=tag2], Left, Top, Width, Height.

- Always returns an empty string, independent if it succeeds or fail.

E.g.:@Display.MainPage.NewPopup(["VALUE = Integer1; NAME = 'Example'"])

Client.OpenDisplay

- Opens a Display. The syntax is as follows:

`Client.OpenDisplay("<DisplayName>")`

- Returns a flag (true or false) indicating success or fail.

E.g.: `@Client.OpenDisplay("About")`

Client.OpenDisplayAtIndex

- Opens Display at x-order. List of Current Display is closed. The syntax is as follows:

`Client.OpenDisplayAtIndex("<DisplayName>", index)`

- Returns a flag (true or false) indicating success or fail.

E.g.: `@Client.OpenDisplayAtIndex("LogOn", 2)`

Client.NewPopup

- Opens a new one. If the Display is already open, a new instance will open. The syntax is as follows:

`Client.NewPopup(input parameters)`

- The accepted input item parameters are:

    LabelList[label1=tag1; label2=tag2], Left, Top, Width, Height.

- Always returns an empty string, independent if it succeeds or fail.

E.g.: `@Client.NewPopup("GAUGE = Integer2")`

Differences

When you add an `OpenDisplay` method to a Solution but then change the name of the Display, you will have some different results if you use the **`@Client`** or `@Display` methods.

- In the **`@Display`** methods: Since the DisplayName is used as one of the properties, the software is smart enough to detect if the Display Name was changed and automatically update the method.
- In the **`@Client`** methods: The DisplayName is used a string type input parameter. If the display name is changed, the user must manually update the method.

NewPopup Characteristics

Symbols

Each copy of a symbol in a display keeps its own parameter values. When the display opens with a parameter whose name equals a symbol parameter name, that value fills the parameter in every copy of the symbol, so the copies show the same tag. To bind copies to different tags, give the open parameter a name no symbol uses, or open one popup per symbol.

The CodeBehind of the popup reads its parameters with `GetLabel` and writes through them with `SetLabel`; see Popup Parameters: GetLabel and SetLabel.

Online Configuration

Online Configuration is another interesting feature to use in a Solution. It allows the modifications that are made to a Solution in the Engineering Environment to affect the application in real time. You can find it in **Runtime Startup**.

In this section...
