Example

Perk/Restore/Swop Programmer

by:	Sebastian Beutel, June 2018
published:	here
description:	Emulate dual programmer: - park programmer - restore programmer - swop programmer

park, restore, swop, programmer

functions

- someusedfunction
- anotherusedfunction

affected properties

- someproperty
- anotherusedproperty

control structures

- anyspecials?

Code

PrkRestoreProgrammer.xml

```
<?xml version="1.0" encoding="utf-8"?>
<avolites.macros>
<!-- Macros to emulate a second programmer - you can store the current programmer,
     retrieve it again, and swop between the current and the stored programmer -->
<!-- As Playbacks.StoreCue() always writes on the current page in the refererenced handle group,
     I chose the Macro window to store these special playbacks. -->
<!-- Sebastian Beutel, 27-06-2018 -->

  <!-- Store programmer into a cue -->
  <macro id="SB.Macros.Programmer.Park" name="SB Park Programmer">
    <description>Park Programmer to a playback</description>
    <sequence>
      <!-- store current programmer to playback #1000 -->
```

```
      <step>Handles.SetSourceHandle("Macros", 1000)</step>
      <step>Handles.ConfirmDelete()</step>
<step>ActionScript.SetProperty("Expert.RecordPlayback.RecordMode.ModeOn
Enter", Playbacks.RecordMode)</step>
      <step>ActionScript.SetProperty.Enum("Playbacks.RecordMode",
"RecordCueModeProgrammer")</step>
      <step>Playbacks.StoreCue("Macros", 1000, false)</step>
      <step>Programmer.Editor.Clear(255, true, false, 0)</step>
      <step>ActionScript.SetProperty("Playbacks.RecordMode",
Expert.RecordPlayback.RecordMode.ModeOnEnter)</step>

      <!-- set usernumber and legend -->
      <step>Handles.SetSourceHandle("Macros", 1000)</step>
      <step>ActionScript.SetProperty("Handles.CurrentUserNumber",
userNumber:10000)</step>
      <step>Handles.SetUserNumber()</step>
      <step>ActionScript.SetProperty("Handles.PendingLegend", "Parked
Programmer")</step>
      <step>Handles.SetLegend()</step>
      <step>Handles.ClearSelection()</step>
    </sequence>
  </macro>

  <!-- Retrieve programmer from a cue -->
  <macro id="SB.Macros.Programmer.Restore" name="SB Restore
Programmer">
    <description>Restore Programmer from a playback</description>
    <sequence>
      <step>Programmer.Editor.Clear(255, true, false, 0)</step>
      <step>Include.SelectPlayback("Macros", 1000)</step>
    </sequence>
  </macro>

  <!-- Swop programmer with previously parked programmer -->
  <macro id="SB.Macros.Programmer.Swop" name="SB Swop Programmer">
    <description>Swop programmer with previously parked
programmer</description>
    <sequence>
      <!-- store current programmer to playback #1001 -->
      <step>Handles.SetSourceHandle("Macros", 1001)</step>
      <step>Handles.ConfirmDelete()</step>
<step>ActionScript.SetProperty("Expert.RecordPlayback.RecordMode.ModeOn
Enter", Playbacks.RecordMode)</step>
      <step>ActionScript.SetProperty.Enum("Playbacks.RecordMode",
"RecordCueModeProgrammer")</step>
      <step>Playbacks.StoreCue("Macros", 1001, false)</step>
      <step>ActionScript.SetProperty("Playbacks.RecordMode",
Expert.RecordPlayback.RecordMode.ModeOnEnter)</step>

      <!-- load previous programmer from playback #1000 -->
```

```
      <step>Programmer.Editor.Clear(255, true, false, 0)</step>
      <step>Include.SelectPlayback("Macros", 1000)</step>

      <!-- delete playback #1000, move playback #1001 to #1000 -->
      <step>Handles.SetSourceHandle("Macros", 1000)</step>
      <step>Handles.ConfirmDelete()</step>
      <step>Handles.SetSourceHandle("Macros", 1001)</step>
      <step>ActionScript.SetProperty.Enum("Handles.OperationMode",
"move")</step>
      <step>Handles.CopyDestination("Macros", 1000)</step>
      <step>Handles.ClearSelection()</step>

      <!-- set usernumber and legend -->
      <step>Handles.SetSourceHandle("Macros", 1000)</step>
      <step>ActionScript.SetProperty("Handles.CurrentUserNumber",
userNumber:10000)</step>
      <step>Handles.SetUserNumber()</step>
      <step>ActionScript.SetProperty("Handles.PendingLegend", "Parked
Programmer")</step>
      <step>Handles.SetLegend()</step>
      <step>Handles.ClearSelection()</step>

    </sequence>
  </macro>
</avolites.macros>
```

Explanation

This explains the functional steps within the sequence. For all the other XML details please refer to Formats and syntax

tbd

How to use it

1. make this macro available

tbd