

Example

# Playback - Set FX Multiplier

<b>by:</b>	Sebastian Beutel
<b>published:</b>	April 2020
<b>description:</b>	Set a playback's Fx Multiplier
<b>remarks:</b>	see also <a href="#">Playback - Set fade-in time</a>

[playback](#), [fx](#), [multiplier](#)

## functions

- [Handles.ClearSelection](#)
- [ActionScript.SetProperty.Double](#)
- [Playbacks.Editor.EnsurePlaybackCueSelected](#)
- [Math.Cast.ToDouble](#)

## affected properties

- [Playbacks.Editor.SelectedPlayback](#)
- [Handles.SourceHandle](#)
- [Playbacks.Editor.Times.CueSpeedMultiplier](#)

## control structures

- [step condition](#)

## Code

[SetPBFxMultiplier025.xml](#)

```
<?xml version="1.0" encoding="utf-8"?>
<avolites.macros>
  <macro id="Wiki.Macros.SpeedMultiplier.025" name="Speed Multiplier
/4">
  <sequence>
    <step>Handles.ClearSelection()</step>
  <step>ActionScript.SetProperty("Playbacks.Editor.SelectedPlayback",
null)</step>

    <step>Handles.SetSourceHandle("PlaybackWindow", 0)</step>
    <step condition="Playbacks.IsCueHandle(Handles.SourceHandle)">
      ActionScript.SetProperty("Playbacks.Editor.SelectedPlayback",
Handles.SourceHandle)</step>
    <step>Playbacks.Editor.EnsurePlaybackCueSelected()</step>
  <step>ActionScript.SetProperty.Double("Playbacks.Editor.Times.CueSpeedM
```

```
ultiplier", Math.Cast.ToDouble(0.25))</step>

    <step>Handles.SetSourceHandle("PlaybackWindow", 1)</step>
    <step condition="Playbacks.IsCueHandle(Handles.SourceHandle)">
        ActionScript.SetProperty("Playbacks.Editor.SelectedPlayback",
Handles.SourceHandle)</step>
    <step>Playbacks.Editor.EnsurePlaybackCueSelected()</step>
<step>ActionScript.SetProperty.Double("Playbacks.Editor.Times.CueSpeedM
ultiplier", Math.Cast.ToDouble(0.25))</step>

    <step>Handles.ClearSelection()</step>
<step>ActionScript.SetProperty("Playbacks.Editor.SelectedPlayback",
null)</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](#)

- The first and last two lines `Handles.ClearSelection()` and `ActionScript.SetProperty("Playbacks.Editor.SelectedPlayback", null)` make handle and playback selection empty so that no other handle/playback is affected by accident.
- `Handles.SetSourceHandle("PlaybackWindow", 0)` selects a playback as source handle
- `<step condition="Playbacks.IsCueHandle(Handles.SourceHandle)">`: only if this is a cue playback handle (and not maybe a macro or group handle which would break the macro)...
- ... `ActionScript.SetProperty("Playbacks.Editor.SelectedPlayback", Handles.SourceHandle)` derive the playback from the given handle (the playback holds the time, not the handle)
- `Playbacks.Editor.EnsurePlaybackCueSelected()` is required for single-cue playbacks to make sure the cue itself is selected/in the editor
- `ActionScript.SetProperty.Double("Playbacks.Editor.Times.CueSpeedMultiplier", Math.Cast.ToDouble(0.25))` finally sets the multiplier to the desired value which needs to be a double, hence the value is cast with `Math.Cast.ToDouble(0.25)`.

## How to use it

- [make this macro available](#)
- while it's well possible to create some macros to set some playbacks to a given multiplier, it does make more sense to integrate this into a bigger context like [Create Workspaces](#) in order to setup a known show environment.

From:  
<https://avosupport.de/wiki/> - **AVOSUPPORT**

Permanent link:  
<https://avosupport.de/wiki/macros/example/setplaybackfymultiplier>

Last update: **2020/04/10 18:27**

