Function

Timelines.ImportMarkersFromString

```
Void Timelines.ImportMarkersFromString(
  Int32 timelineId,
  String importMappingVersion,
  String csvFilePath,
  AcwFrameRate frameRate
)
```

API	https://api.avolites.com/16.0/api/Timelines.ImportMarkersFromString.html
description	Imports markers into the timeline.
namespace	Timelines
parameter	timelineId (Int32): The timeline ID.
	importMappingVersion (String): The XML node for the version of the mapping to use.
	csvFilePath (String): The csv file path.
	frameRate (AcwFrameRate): The framerate to import.
return value	Void

This article was written with massive help from Gregory Haynes; I couldn't have figured this out myself.

There is also the function Timelines.ImportMarkers, however that would expect an XML object as importMappingVersion while this fuction here parses a string which is much easier to handle.

Arguments

- **timelineId** is the identifier or the timeline you want to import the markers into. Unlike for other functions which expect a handle this one expects a titanId. You may retrieve this e.g. from Web API, or you use e.g. Handles.GetTitanIdFromHandle to get the titanId for a specific handle by its usernumber:
  - `Timelines.ImportMarkersFromString(1819, …)` – example with hardcoded titanId
  - `Timelines.ImportMarkersFromString(Handles.GetTitanIdFromHandle("timelineHandleUN=1"), …)` – example to get the titanId from a usernumber
- **importMappingVersion** is a string to be parsed as XML that can define what all the columns are and how to read them.

> Example CSV File:
>
> ```
> externalId,legend,time
> 0,Test,00:00:01.00
> blah,Marker 2,00:01:23.00
> 999,Another Marker,00:12:34.56
> ```
>
> By default it will assume that the CSV file contains headers and it will attempt to use those to figure out which column is which. If it cannot figure out the column headers it will assume the order

> specified above i.e. “externalId”, “legend” then “time”. The external ID can be whatever you want and is used within Titan to detect duplicates i.e. if you import more than once a matching external ID will replace an existing one.

The smallest possible string as importMappingVersion is “`<Version />`” which essentially is “`<Version />`” with the angled brackets written as entities (otherwise they'd be interpreted as XML which would break the macro XML)

> Here is an example of what that string could be (single quotes used instead of double quotes to make it easier to use in a string):

`<Version id='1' name='2020' headerRow='true'><externalId name='#' index='0' /><legend name='Name' index='1' /><time name='Start' index='2' format='' /></Version>` (again the angled brackets need to be written as entities `<` and `>` when using this).

> There is also an XML file included with the personality library (e.g. `C:\Program Files\Avolites\Titan\FixtureLibrary\MarkerImportMapping.xml`) which includes multiple options that can be selected in the UI in the context menu. There is also the provision for a similar file to be in the user fixture library directory.

```
<MarkerImportMappings schemaVersion="1.0"
xmlns="http://avolites.com/MakerImportMapping.xsd">
  <MarkerImportMapping id="reaper" name="Reaper">
    <Version id="1" name="2020" headerRow="true">
      <externalId name="#" index="0" />
      <legend name="Name" index="1" />
      <time name="Start" index="2" format="h':'mm':'ss':'ff" />
    </Version>
  </MarkerImportMapping>
</MarkerImportMappings>
```

> You should recognise the Version section as the bit being passed into the function. Perhaps if the string contained “Reaper”, “Reaper:2020” or “reaper:1” it ought to be able to lookup the appropriate mapping from the library.

- **csvFilePath** is the absolute path (including filename) to the file which is to be imported, e.g. “`C:\Users\{username}\Documents\Titan\Markers\test.csv`”.
- **frameRate** is the framerate to import. This is important to convert frame numbers with different framerates (not tested yet). The API lists AcwFrameRate as Enumeration:

Value	Description
Fps24	24 frames/sec
Fps25	25 frames/sec
Fps29DF	29.97 frames/sec ('drop frame')
Fps30	30 frames/sec
Fps44	internal timecode, 44 frames/sec
Fps100	Winamp and how it is displayed in the Cue View

Value	Description
Fps60	60 frames/sec
Fps1000	Milliseconds

However it looks like you need to use Timecode.ParseFrameRate in order to convert the vaue into a proper framerate object, e.g. `Timecode.ParseFrameRate(100)`.

Example in

Timeline: Import Markers:

```
<step>Timelines.ImportMarkersFromString(
  Handles.GetTitanIdFromHandle("timelineHandleUN=1"),
  "&lt;Version /&gt;",
  "C:\Users\sb\Documents\Titan\Markers\test.csv",
  Timecode.ParseFrameRate(100))
</step>
```

Also used in

- Timeline: Import Markers

Remarks

From:
https://www.avosupport.de/wiki/ - **AVOSUPPORT**

Permanent link:
https://www.avosupport.de/wiki/macros/function/timelines.importmarkersfromstring?rev=1680523795

Last update: **2023/04/03 12:09**