

# XML format

There are dozens of books on XML only - it is a widely used language for many purposes. As always, [wikipedia](#) is a good read. however, based on an example, the (for this purpose) essential items will be explained.

An example macro file might look like this:

```
<?xml version="1.0" encoding="utf-8"?>
<avolites.macros xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Avolites.Menus.xsd">
  <!-- Macro to set PlaybackPaging to never. Sebastian Beutel with help by
Gregory Haynes 05/09/2016 -->
  <macro id="Avolites.Macros.PagingNeverHold" name="Set PbPaging to
NeverHold">
    <description>Sets PlaybackPaging to NeverHold.</description>
    <sequence>
      <step>ActionScript.SetProperty.Enum("Handles.HandlesHeldover",
"NeverHold")</step>
    </sequence>
  </macro>
</avolites.macros>
```

## The 1st line

```
<?xml version="1.0" encoding="utf-8"?>
```

is the opening XML declaration. It is not required but good practice.

## The 2nd line

```
<avolites.macros xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Avolites.Menus.xsd">
```

has some required and some non-required parts:

- < the angled brackets > are essential for every xml tag
- avolites.macros is the essential macro tag which makes Titan interpret this file correctly
- the xmlns... part is the namespace of the file. It is not required and can be omitted.
- this line corresponds with the last line

```
</avolites.macros>
```

. Note the slash at the beginning which makes this a closing tag - **every tag needs to be closed!**

## The 3rd line

```
<!-- Macro to set PlaybackPaging to never. ... -->
```

is a comment, and will not be interpreted by Titan. It is good practice to comment at least what the macro does and who wrote it - and in more advanced macros, comments on the real logic will be of great help.

## The 4th line

```
<macro id="Avolites.Macros.PagingNeverHold" name="Set PbPaging to  
NeverHold">
```

corresponds with line 9

```
</macro>
```

. These two lines hold the macro itself - each file can hold multiple macros where each macro starts with `<macro ...>` and ends with the corresponding `</macro>`.

The parts `id=...` and `name=...` are properties of this macro:

- `id` is required and a unique identifier
- `name` is semi-required and is the name which is shown in Titan - if you omit it, then you'll only see something like 'Macro1'.

## Lines 5 through 8

```
<description>Sets PlaybackPaging to NeverHold.</description>  
<sequence>  
  <step>ActionScript.SetProperty.Enum("Handles.HandlesHeldover",  
"NeverHold")</step>  
</sequence>
```

are the contents of our macro:

- `<description>` corresponds with the closing `</description>`, is optional, and may hold a longer description of the macro
- `<sequence>` corresponds with the closing `</sequence>` and holds the actions which the macro is to perform (one or more steps)
  - it is also possible to use `<start> ... </start>` and `<end> ... </end>` in order to have separate actions when the button is pressed and when it is released, see [Stopwatch/Flash Playback](#)
- `<step>...</step>` is the function which this step is to perform. In this case it calls the function `ActionScript.SetProperty.Enum()` with the arguments "Handles.HandlesHeldover" and "NeverHold".

A brief introduction about possible functions [is available here](#). But essentially, this whole wiki is

dedicated to possible macros 😞

It is possible to bundle some steps together into one block with {curly braces} like this:

```
<sequence>
  <step>Playbacks.SetRecordType("RecordCueModeProgrammer")</step>
  <step>
    {
      Playbacks.StoreCue("PlaybackWindow", 1000, false);
      Handles.SetSourceHandle("PlaybackWindow", 1000);
      ActionScript.SetProperty("Handles.CurrentUserNumber",
userNumber:10000);
      Handles.SetUserNumber();
      Handles.ClearSelection();
    }
  </step>
</sequence>
```

See <http://forum.avolites.com/viewtopic.php?f=20&t=5783>.

#### further readings

- [Introduction to macros](#)
- [Console and simulator](#) - how actions on the consoles are described
- [Recorded vs. coded macros](#) - both kinds: Country, AND Western
- [Macro file format](#) - what to observe when creating macro files
- [Macro Folders](#) - where exactly are the macro files stored
- [Deploying macros](#) - how to import a macro file into Titan
- [XML format](#) - a veeeery basic introduction into the format macro files are written in
- [The Syntax of Functions](#) - understanding how functions are described in general
- [Control Structures](#) - conditions and other means to control the flow
- [Action and Menus](#) - when a menu needs to be toggled in addition to the action
- [Step Pause](#) - a little delay might sometimes be helpful
- [Active Binding](#) - highlighting a macro handle as active
- [Namespaces](#) - a way to keep order of the functions, properties and other stuff
- [Datatypes](#) - numbers, words, yes & no: the various types of values
- [Properties list](#) - the affected system variables of Titan
- [Function list](#) - the functions mentioned in this wiki
- [Examples list](#) - all the contributed macros. And where is yours?

2017/10/13 15:12 · icke\_siegen

From:

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

Permanent link:

[https://avosupport.de/wiki/macros/xml\\_file\\_format?rev=1586689507](https://avosupport.de/wiki/macros/xml_file_format?rev=1586689507)

Last update: **2020/04/12 11:05**



