

# POWER GUIDE

TRACKTION 3



## VST Parameters Structure and .vstxml Files

**MACKIE.**

### Introduction

Welcome to the Tracktion Power Guide Series! This series of guides will give you more insight into some of Tracktion's most powerful, but least understood features. In addition, they will show you how to get more out of Tracktion by providing in-depth descriptions of how to use a particular feature with great results. We hope you enjoy reading these guides and that they will help you become an even more Powerful Tracktioneer.

The Tracktion Team

#### VST Parameters Structure

The VST Parameters Structure is a new addition to the VST 2.4 SDK (Software Developers Kit). Through the use of a .vstxml file, it provides developers with more control over how a VST plug-in presents its parameters to the user. But unlike most aspects of how a plug-in works, the VST Parameters Structure is something that is user-editable and doesn't require a programming degree to do so.

Tracktion 3's support of the VST Parameters Structure and .vstxml files allows the user to do the following with a VST plug-in's parameters:

- Rename them
- Reorganize them
- Group them
- Define Parameter types giving them meaningful discrete values
- Create Short Names that improve readability on control surfaces

The best way to show off this power is with an example. On the next page, you will find a simple .vstxml file for IK Multimedia Amplitube LE that is included with the Tracktion 3 Ultimate bundles.

## IK Multimedia Amplitube LE .vstxml File

```
<VSTParametersStructure>
```

```

<ValueType name="TunerStatus">
  <Entry name="Mute" value="[0.0, 0.25["/>
  <Entry name="Off" value="[0.25, 0.75["/>
  <Entry name="On" value="[0.75, 1.0]"/>
</ValueType>

<ValueType name="WahStatus">
  <Entry name="Manual" shortName="Man" value="[0.0, 0.25["/>
  <Entry name="Off" value="[0.25, 0.75["/>
  <Entry name="Auto" value="[0.75, 1.0]"/>
</ValueType>

<ValueType name="AmpModel">
  <Entry name="Clean" value="[0.0, 0.25["/>
  <Entry name="Crunch" value="[0.25, 0.75["/>
  <Entry name="Lead" value="[0.75, 1.0]"/>
</ValueType>

<ValueType name="CabModel">
  <Entry name="Open Back 1x12" shortName="1x12" value="[0.0, 0.25["/>
  <Entry name="British 2x12" shortName="2x12" value="[0.25, 0.75["/>
  <Entry name="Vintage Closed Back 4x12" shortName="4x12" value="[0.75, 1.0]"/>
</ValueType>

<Param id="0" name="Global Bypass" shortName="Bypass" type="switch" label=""/>
<Param id="1" name="Tuner Status" shortName="Tuner" type="TunerStatus" numberOfStates="3" label=""/>
<Param id="2" name="Gate Level" shortName="GateLev, GatLev" label=""/>

<Group name="Overdrive">
  <Param id="8" name="Overdrive Status" shortName="OD On" type="switch" label=""/>
  <Param id="5" name="Overdrive Level" shortName="DriveLev, OD Lev" label=""/>
  <Param id="6" name="Overdrive Distortion" shortName="Dist" label=""/>
  <Param id="7" name="Overdrive Tone" shortName="Tone" label=""/>
</Group>

<Group name="Amp">
  <Param id="9" name="Amp Model" shortName="AmpMod" type="AmpModel" numberOfStates="3" label=""/>
  <Param id="10" name="Amp Gain" shortName="AmpGain, Gain" label=""/>
  <Param id="11" name="Amp Bass" shortName="Bass, Bass" label=""/>
  <Param id="12" name="Amp Mid" shortName="Mid" label=""/>
  <Param id="13" name="Amp Treble" shortName="Treble" label=""/>
  <Param id="14" name="Amp Presence" shortName="Pres" label=""/>
  <Param id="15" name="Amp Volume" shortName="AmpVol" label=""/>
</Group>

```

## Tracktion 3 and VST Parameters Structure

```
<Group name="Cabinet">
  <Param id="18" name="Cabinet Status" shortName="Cab On, CabOn" type="switch" label=""/>
  <Param id="17" name="Cabinet Model" shortName="Cabinet, Cab" type="CabModel" numberOfStates="3" label=""/>
</Group>

<Group name="FX">
  <Param id="4" name="Wah Status" shortName="WahStat, Wah" type="WahStatus" numberOfStates="3" label=""/>
  <Param id="3" name="Wah Level" shortName="WahLev" label=""/>
  <Param id="16" name="Reverb Level" shortName="Reverb, Rev" label=""/>
  <Param id="22" name="Delay Status" shortName="DlyOn" type="switch" label=""/>
  <Param id="21" name="Delay Level" shortName="DlyLev" label=""/>
  <Param id="19" name="Delay Time" shortName="Time" label="ms"/>
  <Param id="20" name="Delay Feedback" shortName="Feedbk, Feed" label=""/>
</Group>

</VSTParametersStructure>
```

This file may look daunting but it is actually pretty simple. XML uses tags to identify sections, elements, and attributes and if you've ever worked with HTML to design web pages, it will look rather familiar. It is beyond the scope of this document to describe every detail of XML, but we will cover what is necessary to begin using it. For more information, searching for XML in any search engine will yield countless results. A good place to begin is "A Technical Introduction to XML" found here:

<http://www.xml.com/pub/a/98/10/guide0.html>

One of the benefits of XML is that both a human and a computer can read it. So let's start reading it by starting in the middle, at the first Param item.

```
<Param id="0" name="Global Bypass" shortName="Bypass" type="switch" label=""/>
```

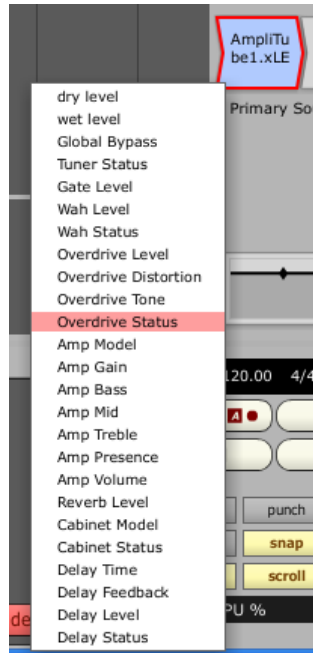
This is called an element, and it begins with a "<" and ends with a ">". Each "Param" element refers to an automatable parameter in the VST plug-in. Following the element name are a number of items called attributes. Each attribute contains a name followed by an equals sign and an associated value in double quotation marks. The first attribute in the Param element is "id," but it should be noted that the order of the attributes within the element is not important. The id attribute is assigned by the plug-in and is essentially the order that the parameter appears in the plug-in's internal parameter list. This is what Tracktion uses to relate your customized parameters to those in the plug-in.

The "name" attribute is the text Tracktion uses anywhere it shows a list of the plug-in's parameters. You can change this to be as long and detailed as you want. After the name, you'll see a "shortName" attribute; this is a shorter name that Tracktion uses for control surfaces. You can enter one or more shortNames separated by commas and Tracktion will use the longest one that will fit on the particular control surface screen. The "type" attribute is used to tell Tracktion more about what the parameter actually does; this will be discussed in detail below.

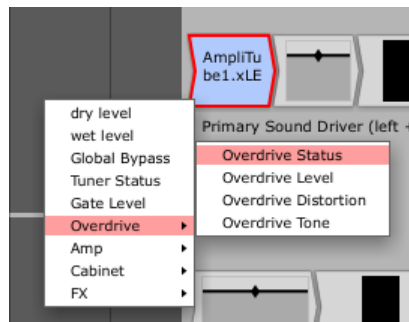
Finally the "label" attribute; it is used as a suffix for the parameter value. So "0.1 ms" could be displayed, for example, where "ms" is the added label. For the Global Bypass element, there is no label as indicated by nothing between the quotes. We could have instead left out the label attribute completely.

Notice that in the example .vstxml file, parameters do not all appear in ID order. They have been rearranged in the list and this causes them to appear in Tracktion's plug-in parameter list in the new order. Also notice that some parameters are contained within a set of Group element tags, one to start the group which has a "name" attribute and one to end it, </Group>. This pair of elements cause the enclosed parameters to appear in a subfolder with the corresponding name in the parameters list. Groups can be nested allowing folders within folders to be created. For plug-ins with many parameters, this can really help with organization.

So, what is the result of creating a .vstxml file for this plug-in? Without the file, the parameters list for Amplitude LE looks like this:



It's a long list without much organization, making it hard to find the parameter you need without having to read through each one. With the above .vstxml file present, Tracktion 3 reformats the list as follows:

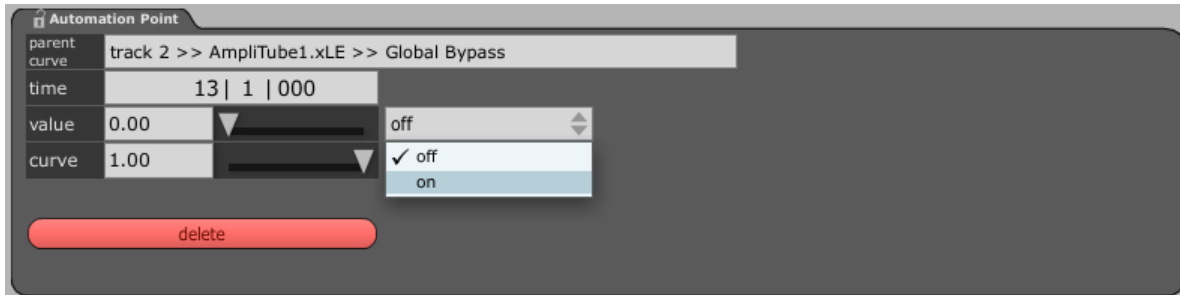


Notice how easy it is to find a parameter because related items are grouped. This also requires less mouse movement, speeding up workflow.

As shown above, the Global Bypass parameter element with ID 0 has a type attribute with a value of "switch". The type attribute tells Tracktion more information about the different possible values for the parameter. Without a type attribute, parameters are displayed and automated as floating point values between 0.0 and 1.0. But these numbers aren't very useful for parameters with discrete values, like a switch. .vstxml allows you to present these kinds of parameters with more logical values. It also allows you to identify a parameter as only having a fixed number of values.

## Tracktion 3 and VST Parameters Structure

The most basic type is the “switch.” It is a special type defined by the VST Parameters Structure to have only two states, On (1.0) and Off (0.0). When you automate a parameter with type switch, Tracktion 3 continues to display the float value in the automation node properties panel, but also shows the corresponding switch state as a drop-down list next to the node value as shown below.



You can use the drop down to set the value of the parameter on or off.

This is great for simple switches, but what about for more complex parameters? For example, the Amp Model parameter in the AmpliTube LE plug-in has three discrete states, none of which are on and off.



Automating these types of parameters with a number between 0.0 and 1.0 is hard because it isn't clear what a particular float value corresponds to; what does a value 0.43 really mean as far as the three position switch is concerned? Again, .vstxml comes to the rescue by allowing you to create your own type attributes for a given plug-in.

The beginning of the example .vstxml file shows these type declarations. Below is the one for the Amp Model parameter:

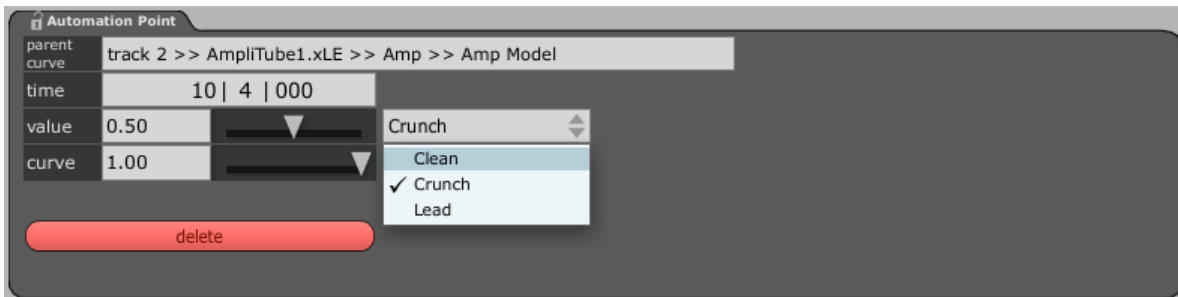
```
<ValueType name="AmpModel">
  <Entry name="Clean" value="[0.0, 0.25["/>
  <Entry name="Crunch" value="[0.25, 0.75["/>
  <Entry name="Lead" value="[0.75, 1.0]"/>
</ValueType>
```

The opening and closing ValueType elements surround the individual elements describing each state of the switch. The ValueType “name” attribute is used to give a unique name for this type of switch. Each Entry element also has a “name” attribute that is used to give the particular discrete parameter value a name. The “value” attribute defines the range of floating point values that correspond to that name. A bracket facing the number means inclusive while a bracket pointing away from the number means exclusive. So the first range of 0.0-0.25 (including 0 and excluding 0.25) sets the switch to Clean, the second range of 0.25 to 0.75 (including 0.25 and excluding 0.75) sets the switch to Crunch, and the third range of 0.75 to 1.0 (including 0.75 and 1.0) sets the switch to Lead.

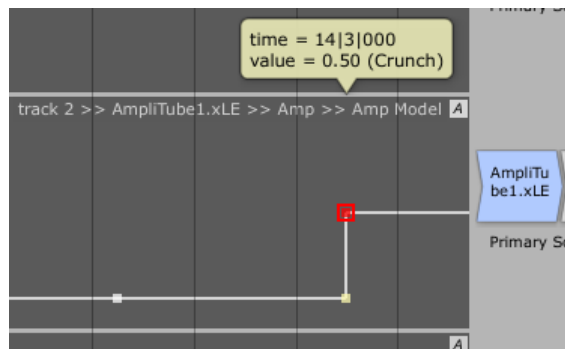
But we're not done yet; this is a two step process. The above code defines the value range, but we haven't yet identified what parameter uses it. This is done in the corresponding parameter element.

```
<Param id="9" name="Amp Model" shortName="AmpMod" type="AmpModel" numberOfStates="3" label="" />
```

Notice we have used the ValueType element name attribute value (AmpModel) as the value for the Param type attribute. This tells Tracktion 3 that parameter id 9 is of type AmpModel. The next attribute, "numberOfStates," defines that there are only three states to this parameter. In almost every case where you use a custom type, you will also define the corresponding number of states. By doing this, Tracktion now knows how to divide up the floating point range of values defined in the ValueType element for the parameter. Like the switch type, Tracktion presents these three settings to the user as a drop-down list in the automation node properties panel as shown below:



When using switch or custom types, Tracktion also makes it easier to edit automation by dragging the automation nodes in a number of ways. First, when dragging the node it will only snap to the number of discrete steps making it easy to find a value. Second, the pop up balloon showing information about the node will display the name from the corresponding ValueType entry as you drag:



### Creating .vstxml Files

Now that you have seen the power of .vstxml, you'll probably want to begin creating your own and sharing them with others. To create a .vstxml file for a plug-in, all you need is a text editor and some time. But there is a tool that makes starting your file much easier.

Go to the following link and download the VST 2.4 SDK from Steinberg Media Technologies GMBH:

<http://www.steinberg.net/331+M52087573ab0.html>

## Tracktion 3 and VST Parameters Structure

Once downloaded and unzipped somewhere on your hard disk, you can find an application called “vstparamtool.exe” at the following path in the SDK folder:

```
\vstsdk2.4\bin\win (PC users)
\vstsdk2.4\bin\mac (Mac users)
```

Run this tool, choose “File>VST Plug-in>Extract Parameters...” and locate the plug-in you wish to create a .vstxml file for. The tool will load the parameters, one per line. Choose your text editor from “File>Select XML Editor...” You don’t need anything special; Windows Notepad or Mac TextEdit will do. Click the “Edit” button and a text file will open with the parameters listed in their default order. Then all you need to do is edit the file using the methods you learned above. Before you edit too much, you may want to save a copy of the original .vstxml file with a different name in case you need to go back. Once you get the hang of it, this won’t be necessary, especially because the tool has a nice feature that allows it validate your xml to make sure you don’t make a mistake. Once you have made your changes to the .vstxml file, save it and choose “File>Refresh”. If the tool doesn’t give you an error, your xml file is formatted properly. If it does show an error pop up, go to the line number indicated and figure out what went wrong. It’s usually something simple like forgetting to close a quote or misspelling an attribute.

Once completed, on the PC, place the finished file in the same folder as the plug-in .dll file. On the Mac, put it in the Contents>Resources Folder of the plug-in bundle by right-clicking on the plug-in file and choosing “Show Bundle Contents.” Once you are complete, relaunch Tracktion 3 for your changes to take effect for that plug-in.

## Conclusion

This document is simply an introduction to the VST Parameters Structure and how it works with Tracktion 3. We hope it gets you excited to create your own .vstxml files for your most used plug-ins and to share them with your fellow Tracktion users.

For more information, please see the VST 2.4 SDK documentation and specifically, the VST Parameters Structure document found at the following location in the SDK folder:

```
/vstsdk2.4/doc/html/vstparamstruct.html
```

Written by Ben Olswang  
Product Requirements Manager  
LOUD Technologies Inc.

“Mackie” and the “Running Man” figure are trademarks or registered trademarks of LOUD Technologies Inc. All other brand names mentioned are trademarks or registered trademarks of their respective holders, and are hereby acknowledged.

Part No. SW0521 Rev. A 06/07  
© 2007 LOUD Technologies Inc. All Rights Reserved.



# POWER GUIDE

TRACKTION 3



16220 Wood-Red Road NE • Woodinville, WA 98072 • USA  
United States and Canada: 800.898.3211  
Europe, Asia, Central and South America: 425.487.4333  
Middle East and Africa: 31.20.654.4000  
Fax: 425.487.4337 • [www.mackie.com](http://www.mackie.com)  
E-mail: [sales@mackie.com](mailto:sales@mackie.com)

**MACKIE®**