CUSTOM ORGAN DESIGN MODULE

VERSION 7.0

HAUPTWERK VII

VIRTUAL PIPE ORGAN

milan DIGITAL AUDIO

Custom Organ Design Module User Guide for the Hauptwerk Virtual Pipe Organ

Hauptwerk version 7

What it is and what it's for

Very briefly, the Custom Organ Design Module exists primarily to allow end-users and amateur/hobbyist/semi-professional sample set producers quickly and easily to create high-quality organ definitions that can be loaded into Hauptwerk for personal use or for distribution as part of sample sets, and that can take good advantage of Hauptwerk's features and give excellent results.

The next three sections give some background and explain why this is useful.

Sample sets and their data

Hauptwerk allows different sample sets to be installed and loaded, providing virtual models of many different pipe organs and similar instruments. A good variety of such sample sets are available commercially.

Sample sets are comprised of some or all of the following types of data:

1. Audio samples, usually one or more for each pipe on the organ, plus action noises, effects, etc.
2. Special tremulant waveform samples, usually provided at regular intervals for each pipe rank affected by a tremulant, describing how the pitch, amplitude and harmonic content of the pipe(s) vary with time when the tremulant is active.
3. Images used to display the virtual console and other aspects of the organ and to allow the user to interact with it with a computer screen, such as a touch-screen, or the mouse.
4. The organ definition file.
5. An HTML or PDF file and any associated components, used to display information about the instrument and documentation relevant to it.

The creation of a large professional-quality sample set often involves many months of work, mostly in the preparation of the pipe samples, but a very significant amount still being required for the remaining parts.

Native organ definition files

A full native Hauptwerk *organ definition file* is a text file in XML format, the contents of which define the *organ definition database*, being a normalized relational database which provides all of the information needed by Hauptwerk to model an organ. For example, each pipe is listed individually, describing such intricate details as how the air flow rate through it varies as the pressure across it changes, how its speech changes in character as flow rates vary, which samples are to be used for it under which conditions, how its characteristics change with each available tremulant and enclosure, and so on.

The full native organ definition format is extremely large and complex, of necessity so that Hauptwerk is able to provide a sufficiently detailed and generic model for (almost) all types of organs and similar instruments that exist. Although often technically possible, it would not normally be feasible to edit a native full organ definition file manually in a standard text editor because of the very large number of related tables, records and their settings, which must be updated to accomplish a given change, as well as its overall size.

Generation of an organ definition using custom-developed third-party software and/or editing it via dedicated XML manipulation software are the methods most commonly used in the production of complex commercial sample sets for Hauptwerk since they give the greatest possible degree of flexibility and control.

However, even with the benefits of such tools, the task of defining an organ definition is still very involved, and it often requires several weeks of work to produce such a definition for even a small or medium-sized complex sample set. A reasonable understanding of pipe organ construction and some of the relevant physics principles involved in the Hauptwerk models is also required in order to configure parts of the organ definition database to best effect, and it often takes several months to master Hauptwerk's full native organ definition format.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

It is thus expected that most users and amateur/hobbyist/semi-professional sample set producers would not have the time or inclination to study the full organ definition format, organ building, physics principles, or to devote to the configuration of the database. Because no user settings are stored within the organ definition database, it is also intended that no user should ever need to examine or edit the organ definition databases in order to use and appreciate any sample set fully within Hauptwerk.

In summary: Hauptwerk's full native organ definition file format is enormously flexible and powerful but correspondingly complex. It is not aimed at end-users or amateur/hobbyist/semi-professional sample set producers, so very few people would need to understand or use it.

The Custom Organ Design module

Despite the complexities of the full organ definition format, many users wish to create their own organ specifications using parts of sample sets that they have installed (where the licensing of those sample sets allows it), tailored to their own particular musical tastes and the physical organ consoles they have available from which to play them. There is thus a need to be able to define and adjust organs simply and quickly. Amateur/hobbyist/semi-professional sample set producers also require a simple organ definition format to work with that provides sufficient flexibility to make a fully-functional organ of reasonable complexity and to cater for functionality commonly found on most types of organs, but that remains quick and easy to learn, manipulate and use.

Hauptwerk's Custom Organ Design Module is intended to provide for those needs.

It provides a simple alternative and parallel organ definition format with which to define an organ easily: the *custom organ definition format*. The custom organ definition format is another XML text file format, the contents of which are termed the *custom organ definition database*. The file format is simply referred to as a database because the data within it are organized into tables and fields; it is still just a simple XML/text format, and does not need any third-party database software to edit it.

The format has been very carefully designed so that an absolute minimum of information must be given to define an instrument, whilst still allowing large and relatively complex instruments to be modeled (including both classical and theatre organs), and for most of Hauptwerk's features to be used very effectively. No understanding of the physics of the models is required, and a minimal understanding of pipe organ building is necessary.

It is also designed in such a way that very few data tables and records are required, so that the files remain small and easily-managed, and so that it is an easy task to configure an instrument using simply a text or XML editor.

When Hauptwerk loads a custom organ definition file, it first 'compiles' the data contained within it to its native full organ definition format, saves the resulting organ definition file, then finally loads that file as normal. During the compilation process the full organ definition is generated using the settings provided within the custom organ definition file, with sensible defaults being calculated automatically for most of the complex settings in the full organ definition (such as the air flow rates within the various parts of the organ), in order to keep the custom organ definition format simple.

The resulting compiled organ definitions can be used in the same way as any other native organ definition files; they can be loaded as normal via the *Organ* menu in Hauptwerk, or shared with other Hauptwerk users, whether or not those users are familiar with use of the Custom Organ Design module itself. The role of the Custom Organ Design module is thus simply that of a compiler, taking as an input a small and easy, yet powerful and flexible, organ definition format, and producing as output a full native Hauptwerk organ definition file.

The Custom Organ Design Module allows for fully-customizable graphical virtual consoles to be created and has support for most functional features found on both classical and theatre organs. Hence it is an ideal tool with which amateur/hobbyist/semi-professional sample set producers can create organ definition files suitable for public release as part of sample sets, as well as for personal use. Sample set producers that require more flexibility still can use the Custom Organ Design Module to generate an initial organ definition, and then fine-tune the compiled native organ definition further if required. Once a custom organ definition has been loaded/compiled, a sample set producer simply needs to edit the generated native organ definition file to change the unique organ ID in order to make it suitable for distributing publicly as part of a sample set. Please contact us to be assigned the necessary ID if you wish to make a sample set or native organ definition file available publicly, since the values *must* be unique globally, as covered later in this guide. (There is no charge for the service.)

Please note that the custom organ definition format still uses text/XML files; there is not yet a graphical interface for the organ design process itself. Hence we would generally advise novice computer users to start with simple custom organs and examples first.

Licensing

The Custom Organ Design Module is included with all editions of Hauptwerk. No separate license is required for its use. However, it is only possible to take full advantage of some organ features that optionally can be modeled with the Custom Organ Design Module using the Advanced Edition, such as an organ wind supply model or virtual consoles designed for multiple touchscreen use. Also, no technical support is available for the Custom Organ Design Module in the Hauptwerk Lite Edition (although the module may still be used within it).

Support

Although it is quick and simple to create simple custom organ definitions, it also allows quite complex and sophisticated organ definitions to be created. The design goal has been to keep the format as small and simple as possible, but inevitably some familiarization is required, and there is a certain amount of additional complexity if you wish to take advantage of more advanced features. Hence we would generally advise novice computer users to start with simple custom organ definitions and examples first.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Important note: because of the sophistication possible, and because of the 'DIY' nature of creating or editing organs to taste, it needs to be emphasized that we are sorry that we cannot provide personal tuition, or assistance in creating/editing custom organ definitions, to end-users. This user guide and an extensive set of examples are provided to help. We will usually only be able to provide a small amount of support beyond that. However, you can of course exchange help and advice with other users via our website forum.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Basic operation and getting started

A custom organ definition is loaded into Hauptwerk using the *Design tools* | *Load custom organ ...* menu function. That's the only menu function needed to use the Custom Organ Design Module.

Hauptwerk's installer installs several example custom organs for you to get you started, all of which use samples from the St. Anne's, Moseley organ sample set installed with Hauptwerk. If you haven't used the Custom Organ Design Module before, start by loading one of these examples:

- Use *Design tools* | *Load custom ... organ* to select the entry named *ExampleCustomOrgan2* and click *OK*.
- The *Load Organ Design Options* window will appear. Leave all the settings at their defaults and click *OK*.
- The Custom Organ Design Module will now compile the custom organ definition to a full native organ definition file and load it as normal, presenting you with the *Rank Audio/Memory Options and Routing* screen as usual the first time that you load an organ. Click *OK* (or disable some ranks first if you have less than 1 GB of memory).
- Once the organ has loaded, try using and playing it as normal. This should give you an idea of what can be achieved using the module, and is a relatively complex organ intended to serve as an example for various complex features such as Pizzicato couplers and a Thunder piston, and also showing a custom graphical console using some of the control images from the standard library supplied with Hauptwerk.
- When you have finished playing the organ, select *Organ* | *Load organ ...* from the menu. You will see that an entry named *ExampleCustomOrgan2* now exists. This is the compiled full native organ definition that the Custom Organ Design Module generated from the custom organ definition when you loaded it via the *Design tools* menu. If you wish you can now load that organ using the functions on the *Organ* menu as you would with any other organ. However, if you subsequently made any changes to the custom organ definition then they wouldn't have any effect until you re-loaded the custom organ definition using *Design tools* | *Load custom organ …* again, since only by doing that will the Custom Organ Design Module be invoked to regenerate the native organ definition.
- Now use *Design tools* | *Load custom organ ...* to select and load the *ExampleCustomOrgan1* entry, clicking OK on the following options screens as before.
- When the organ has loaded try it out and look at the various display page tabs. This organ is a very simple example that just uses the standard virtual console display that the Custom Organ Design Module generates automatically by default. Creating such an organ with a standard display is very quick indeed (probably considerably less than an hour) once you have mastered the custom organ definition format.
- Now use Finder (if you're using macOS) or File Explorer / Windows Explorer (if you're using a Windows PC) to navigate to the *CustomOrganDefinitions* folder inside the *HauptwerkUserData* folder. If you performed a default Hauptwerk installation, that folder will be found inside */Hauptwerk* (macOS) or *C:\Hauptwerk* (Windows).
- Open the file named *ExampleCustomOrgan1.CustomOrgan_Hauptwerk_xml* inside a text editor, such as macOS's */Applications/Utilities/TextEdit* or Windows' Notepad. This is the custom organ definition for the organ that you just loaded. Browse around the file and compare its sections ('tables') objects ('records') and individual settings ('fields', 'attributes' or 'parameters'; used interchangeably as terms) with the reference documentation at the end of this guide.
- If you want to try changing it, you need to make a copy of the file, rename the copy, open that copy in your text editor, and change the *UniqueOrganID* setting near the top of the file to a new unique value above 800010 (so that it doesn't match any of the example organs). You **must** always make sure that the organ ID is unique, otherwise settings, voicing and combination files from one organ will overwrite those of another.
- You can then try making changes to your copy and loading it using *Design tools* | *Load custom organ ...* to see what happens.
- Use the '*Design tools* | *View Custom Organ Design Module … format documentation'* menu function as reference for the format.

Now that you have a basic understanding of operating the module, try making some custom organs yourself, referring to the rest of this guide for more details, and also to the other example files.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

The example files

Often the quickest way to learn something is to see examples of it in action. Hence we've tried to provide a good selection of example custom organ definition files to get you started and to be used as reference.

A total of six example organ definitions are provided, ranging from a very simple organ with an automatically-generated virtual console to a large complex organ with a custom graphical display, action noises, and demonstrating how to implement many other advanced features, such as pizzicato. Four versions, increasing in sophistication, of the St. Anne's, Moseley organ are included in those examples, both to show the process of building up an organ definition and to allow easy customization of the sample set by users.

All of the files can be found in the *CustomOrganDefinitions* folder (see the last section for its location), and all custom organ definitions must be stored in that folder in order to be usable in the Custom Organ Design Module. Custom organ definitions have the file extension *....CustomOrgan_Hauptwerk_xml* and can be edited in a text editor or XML editor.

The purpose of each example file is as follows:

- **ExampleCustomOrgan1**: This is a very simple custom organ that demonstrates how easy it is to create a functional organ quickly. It uses the default standard generated virtual console display, rather than a custom graphical virtual console, which makes it very quick to create and edit, with minimal settings to change. It has no action noises and uses stop, coupler and enclosure behavior which is determined by the stop, coupler and enclosure codes. Such behavior is sufficient for the vast majority of classical organs.

- **ExampleCustomOrgan2**: This is intended to demonstrate as many functional complexities as possible. A custom graphical console is used (see below), but with only one display page tab and screen layout. The organ also has key action noise, stop action noise and blower noise. A multi-rank stop is included (the Sesquialtera) as well as various borrowed/unified ranks. An array of standard divisions and couplers and non-standard divisions are demonstrated, as well as complex custom couplers (Swell Reeds to Choir) and couplers with special actions (Great to Choir Pizzicato coupler). Reversible pistons are demonstrated, as are complex effects pistons (Thunder). Internal couplers that copy their states from others are also included for the Swell division to allow the Swell Reeds to Choir to be implemented whilst the Swell keyboard can play both its reed stops and other stops, even though they are attached to different divisions internally. Most of these features you will probably never need to use. However, this example organ definition does serve one other very important function: the *CustomDisplayControlStyle* and *CustomDisplayKeyboardStyle* sections/tables contain a ready-made master library of display styles that you can copy and paste into your own custom organ definitions.

- **ExampleCustomOrgan3-StAnnes-Simplified**: This is the St. Anne's organ recreated in the Custom Organ Design Module in the simplest possible way. There are no custom graphical displays, no noises and no additional pistons or other gadgetry. You would normally start by creating and fine-tuning an organ definition in this way before complicating it with graphics or other 'frills'.

- **ExampleCustomOrgan4-StAnnes-WithBasicGUIConsole**: Beginning the process of building up a fully-fledged organ for inclusion in a sample set, this adds a single custom graphical console display page and reversible pistons to example 3.

- **ExampleCustomOrgan5-StAnnes-WithBasicGUIAndNoises**: Building on example 4, this adds key action noise, stop action noise, tremulant action noise (two types) and blower noises.

- **ExampleCustomOrgan6-StAnnes-WithAllFrills**: Finally, this adds additional graphical console display pages to example 4, making it suitable for use on dual-touchscreen computers, whilst also retaining the overview console display page for those that want to use it on single-monitor systems, or who don't have the Advanced Edition of Hauptwerk (only the Advanced Edition supports multiple monitors). It also includes an alternate screen layout for the left and right stop jamb display pages, thus natively supporting both landscape and portrait orientations for touch-screen monitors (Hauptwerk selects the most appropriate one automatically, depending on the size of the window/monitor).

Important note 1: please do not edit any of these example files directly, since you might want to refer to them later, and they will be overwritten next time that you run Hauptwerk's installer. Instead, make a copy of them, rename them and change the unique organ IDs in the copies, as described in the last section. Then make any edits you wish in those copies.

Important note 2: example custom organ 2 contains a ready-made master style library, ready for you to copy and paste into your own custom organ definitions if you want to use the library of graphical control and keyboard images supplied with Hauptwerk.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

An overview of the objects in a custom organ definition and characteristics of organs created with the Custom Organ Design Module

Since the intention of the module is to make it as quick and easy as possible to configure functional organs, some aspects of the compiled organ definition are fixed, or calculated automatically according to fixed rules.

The organ overall has a 'master combination capture' (setter) switch, a general cancel piston and set of 20 native general combinations. All combinations pistons allow capture.

Any or all of seven standard fixed divisions may be included for general use (six manuals and a pedalboard), and an enclosure (swell box) can optionally be assigned to each. A fixed set of choices exist for the available enclosures; at most one per division, and optionally also a general swell. Each division has its own keyboard, divisional cancel piston, set of 20 divisional combinations and key actions. Stops, couplers and tremulants can be assigned to any of these divisions and are stored and recalled within the divisional combinations for the division to which they are assigned.

For each standard division, a parallel-rise weighted wind reservoir, wind-chest, regulator valves and indicators can optionally be created. Almost all of the wind model is calculated and generated automatically, with just a very small number of the most important parameters being provided to adjust its behavior, and an option to disable it. Any standard division may be used as a source of wind for ranks.

Additional (non-standard) divisions are allowed (up to sixteen in total, including the seven standard divisions), but they would normally only be used for implementing special couplers, such as 'Choir Reeds to Great', where an additional division would need to be used for the Choir reeds so that could sound from the Choir or the Great keyboards. These additional divisions do not have their own divisional combinations or wind-chest. Any stops, couplers and tremulants attached to them must also be associated with one of the seven standard divisions to define in which divisional combinations they should be stored and recalled. In the preceding example, stops attached to the special 'Choir reeds' division should be associated with the 'Choir' division for combinations, so that the main Choir division's divisional combinations store and recall their states.

A coupler is normally specified by selecting it from a fixed range of choices, determined by its 'coupler code', which defines which divisions it links and at what pitch. Those standard codes only allow coupling to/from the seven standard divisions. However, a range of 'custom coupler' codes exists to allow for complex coupling, such as pizzicato, or coupling at mutation pitches. The coupler code always defines the division in whose divisional combinations its state will be stored/recalled, which can only be one of the standard seven divisions. To couple to/from other divisions, the source and destination divisions can optionally be overridden to any division, including the remaining non-standard divisions.

Tremulants similarly may be selected from a fixed range of choices. As with couplers, the 'tremulant code' defines the division with which it is associated for divisional combination purposes, but has no other effect, since any rank can optionally be associated with any tremulant, regardless of the tremulant's division.

Pipe samples may be used from any sample set (if the license of the sample set permits it) to form a rank, and tremulant waveform samples from any sample set may be applied to it (also subject to sample set license). One or more ranks, or parts thereof, can be assigned to each stop, allowing unification/borrowing between stops and divisions. There is a many-to-many relationship between ranks and stops, so a stop can cause several ranks to sound at once and/or several parts of ranks and/or a rank may be used by several stops (unification).

The *Rank* table is by far the most complex in the custom organ definition database, and defines how all of the pipework should be generated - the samples to use, their original pitch, the pitch at which they should sound (when played from a unison-pitch stop), voicing, modulation by the wind model and tremulants (pitch, amplitude and harmonic content), the acoustical effects of any enclosure, which tremulant waveform samples should be applied (if any), and so on. Although the table requires a reasonable amount of configuration, all such complexity is concentrated in a single record for each rank - there is no need to specify values for each pipe - and the options are very much simplified over the full organ definition database, whilst still allowing a great deal of flexibility where it is most needed.

As with couplers, the 'stop code' of a stop defines the division in whose divisional combinations its state is stored/recalled, which can only be one of the seven standard divisions. Its source division (the division to whose key action it will be attached) is also determined by default from that code. However, to allow use of the non-standard divisions it is possible to override the source division, as with couplers.

Various additional options exist for stops, couplers and tremulants to allow special/unusual behavior. For example, they can optionally be non-displayed, non-latching (momentary, e.g. a theatre organ 'toy counter' effect), default to engaged (e.g. a blower 'stop'), not directly accessible to the user via MIDI, and need not appear in divisional or general combinations. Any stop, coupler or tremulant may also be set to track the state of any other. This is useful for 'internal' objects that are used to implement complex cases, usually in combination with making the object non-accessible via MIDI, non-displayed and not appearing in combinations. As an example, a visible and accessible coupler could actually trigger two couplers internally to connect a source division to two destination divisions to allow for a 'Swell Reeds to Choir' coupler. In that case a 'Swell Octave' coupler would need to play both the Swell division and the Swell Reeds division from the Swell keys, requiring two couplers.

The 'example custom organ 2' example demonstrates most of these complex cases for reference, whilst the 'example custom organ 1' example demonstrates simple standard cases. For most classical organs these types of special behavior would not be needed.

Reversible and other 'short-cut' pistons may also be defined to allow a (reversible) piston to toggle the state of stop, coupler or tremulant or to allow the piston to turn on the target whilst held in. Theatre organ 'toy counter' effects could also be implemented in that way.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Object codes, references and object names

Most types of object in the custom organ definition format are identified by a code. For example, the *Coupler* table has a *CouplerCode* attribute which defines the coupler code of each entry in the table. The code serves several purposes:

- It identifies the object uniquely. No two objects of a given type can have the same code. When you need to refer to the object from a different type of object, you then just need to specify its code. For example the *TremulantCode* attribute of the *ShortcutPiston* table refers to the *Tremulant* table, so to tell the Custom Organ Design Module that the piston should trigger a particular tremulant you need to set that piston's *TremulantCode* value to match the corresponding value from the *Tremulant* table for the tremulant you want it to trigger.
- It defines its primary 'role', which Hauptwerk uses to help choose appropriate actions when a user right-clicks on it on the virtual console. For example if an *Enclosure* object has the code 220 then that tells the Custom Organ Design Module that it is the swell box for the Swell division.
- For some types of objects, such as *Stop* and *Coupler* objects, it determines the division(s) to which it will be connected functionally (although it is possible to override the division(s) for special cases).
- For *Stop*, *Coupler* and *Tremulant* objects it also determines the division in whose divisional combinations its state will be stored and recalled.

Most tables also have a *Name* attribute. The value you specify for the name never has any effect on the functioning or structure of the organ. It is primarily there to help you, as the designer of the custom organ, be able to identify the object in user-friendly way. The *Name* setting is also usually displayed to the user via the settings screens on the *Organ settings* menu in Hauptwerk, the contents of which are sorted alphabetically by the object names. Thus you should make sure you use names that will be meaningful to the user of the organ.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

The default standard console display

By default a standard, fixed virtual console display is generated. Four pages (tabs) are created on the console screen. The 'keyboards' page shows the manuals, pedalboard and any expression pedals for enclosures. The 'controls' page lists all of the stops, couplers and tremulants on the stop jambs for the standard divisions. A maximum of 38 stops, couplers and tremulants in total can appear for each division. If more exist, then the remaining items will not be shown. The jambs are simply presented as lists with a column for each division, and they are automatically sorted according to the function code (see above) of each object. The twenty divisional combinations and the divisional cancel piston are displayed in a column to the left of each division's jamb. The twenty general combination pistons, along with the general cancel and setter piston, are displayed in a row along the bottom of the screen.

All standard divisions are displayed in columns on the 'wind' page, organized by division. Non-standard divisions do not appear on the default standard generated 'keyboards', 'controls', or 'wind' display pages.

The 'enclosures' page displays indicators for the positions of the expression pedal and shutters for each enclosure, and optionally also the wind pressure inside the swell box (the wind model can be enabled or disabled for an enclosure).

Generation of the default standard display pages can be enabled or disabled individually as required. Those if using the Custom Organ Design Module to produce sample sets for public release you might want to disable the standard display pages in favor of an aesthetically-pleasing custom display.

The custom display functionality

As well as the default standard virtual console display, you can optionally create a fully-customized graphical display using images from a standard library we provide, or using images of your own. Any image file can be used for any virtual control (stop, coupler, tremulant, piston, keyboard, etc.) and image files can be specified freely for the console screen backgrounds. A toolkit of useful ready-made control and background images is included with Hauptwerk, including various styles of draw-knobs, pistons, theatre organ tabs, rocker switches, labels, expression pedals, keyboards and pedal boards. Most are specifically designed to be suitable for touch-screen use. These make it easy to create a graphical console quickly.

Multiple custom display page tabs may be used for compatibility with Hauptwerk's multi-monitor/multi-touchscreen support (available in the Advanced Edition). Any virtual control may be displayed using custom images on up to two page tabs simultaneously. For example a draw-knob could appear on a console overview page and also on a page representing the left stop jamb for use with two touchscreens. The *CustomDisplayPage* table is used to define the page tabs and their background images.

Images, fonts, keyboard key images and layouts and other display properties are defined as a library of 'styles' so that only a few parameters need to specified for each virtual control. An extensive ready-made style library is supplied in the 'example custom organ 2' custom organ definition file, ready for you to copy and paste into your own organ definitions and use immediately. The *CustomDisplayControlStyle* and *CustomDisplayKeyboardStyle* tables define the style library. To make use of our pre-defined styles and images, simply copy and paste those two tables from our 'example custom organ 2' custom organ definition into your own custom organ definition file. You can then add more of your own if you wish, or delete those that you don't want to use (which slightly saves memory and reduces organ loading time).

For each virtual control you then just select the style, specify its X and Y co-ordinates in pixels and the text (if any) to be displayed upon it. The *CustomDisplay_...* fields of the *Stop*, *Coupler*, *Tremulant*, *Enclosure*, *Division*, *_General, ShortcutPiston* and *CustomDisplayLabel* tables provide that functionality. In each case there is a set of fields prefixed *CustomDisplay1_...* and another identical set prefixed *CustomDisplay2_...* . Either, neither or both may be used, thus allowing any control to appear simultaneously on up to two different custom display pages.

Divisional and general combinations are always displayed in rows by specifying the number that should be displayed and the X and Y co-ordinates of their first and last displayed pistons. The rows need not be horizontal.

The *CustomDisplayLabel* table allows static (non-functional) image items (usually labels) to be positioned freely on any custom display page.

Several additional *CustomDisplay_...* fields exist on the *_General* table that allow the size of the virtual console to be defined, whether the default standard virtual console display pages are enabled, and which custom display page tab should be selected by default when the user loads an organ for the first time. (For subsequent loads the last-selected page is remembered.)

Additionally, an alternate layout may be specified for some or all custom display pages, defined with the *..._AlternateLayout1_...* and *..._AltLayout1_...* fields on various tables. A separate (global) size may be specified for this layout (within the *_General* table) and each custom display page must then specify whether it includes the alternate layout or not. This allows an organ to include up to two custom layouts natively laid out optimally for different screen orientations or aspect ratios for any given display page. Typically the main layout would be used for landscape views of the console pages and the alternate layout for a portrait views, to allow for portrait-oriented left and right stop jamb touch-screens. For any display page the user selects Hauptwerk automatically chooses the layout that most closely matches the aspect ratio and/or size of the virtual console window on which it's being displayed. By default it will zoom the chosen page (aspect ratio) to fit as best it can. You can see this in action by loading the *ExampleCustomOrgan6-StAnnes-WithAllFrills* example custom organ in Hauptwerk, selecting the left or right jamb tab at the top of the virtual console screen, and then dragging the main Hauptwerk window between a wide shape (landscape) and a tall and narrow (portrait) shape. The virtual stop jamb should toggle automatically between landscape and portrait layouts. Note that all controls that appear on any given display page that includes the alternate layout must also appear on the alternate layout for that page. Thus an alternate layout does not allow the controls to be reorganized across display page boundaries.

Using the custom display functionality does increase the time it takes to create a custom organ but allows a very professional-looking virtual organ to be created which could be suitable for release publicly as part of sample set.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Creating your own custom display styles and images

A control style is a set of images that can be used to display any stop, coupler, tremulant, piston, label, static image object or swell pedal.

As covered above, a standard ready-made library of styles can be found in the 'example custom organ 2' custom organ definition, ready to copy and paste into your own custom organ definition file. However, it is also possible to use your own images by creating entries in the *CustomDisplayControlStyle* and/or *CustomDisplayKeyboardStyle* tables.

Image files can be stored in *.png, .bmp*, *.tiff* or *.jpg* formats and must be located within an installation package folder (see below), from which they can be referenced. All images that will be used to display different states of a given control, need to be listed together as 'stages' for the style in the *CustomDisplayControlStyle* table, and they must all have the same size in pixels. The *Image_Stage00_ImageFilename* setting defines the image to be used for the 'off' state for switches (stops, couplers, pistons, etc.) and expression pedals. *Image_Stage01_ImageFilename* defines that for the 'on' state for switches. For expression pedals, up to sixteen stages can be specified, and Hauptwerk will map the number specified across the full possible range of expression pedal positions. For labels and other static image objects, only the *Image_Stage00_ImageFilename* setting is used. A mask may optionally be specified for a control style (applying to all stages). In a mask, white indicates transparency, black opacity. (Semi-)transparent PNG files are also supported directly, requiring no separate mask file, and are thus the recommended format.

Sets of keyboard key images may be defined using the *CustomDisplayKeyboardStyle* table. As with control images, a mask may optionally be specified for each key shape.

To use a custom image file for the background of a virtual console display page, its filename and installation package are simply specified directly in its *CustomDisplayPage* entry.

Please note that all entries in the display style tables cause images to load into memory, and thus it is best to eliminate unnecessary entries when you have finished working on an organ definition in order to save memory and (slightly) minimize organ loading time.

Images files are stored within the sample set data cache (covered later).

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Ranks and audio samples

Hauptwerk uses audio samples in Microsoft WAV format (or the custom copy-protected .hbw format) to produce its pipe sounds. Typically, one or more samples is used per pipe, but samples can also be provided at any note interval. Samples may be referenced from any Hauptwerk sample set installed on the computer provided that they adhere to the requirements for compatibility with the Custom Organ Design Module and that the license for the sample set allows it.

The method by which samples are referenced is covered later in this guide.

Hauptwerk's *Rank* table defines both the pipes and ranks in a custom organ. Although noises and effects are not strictly pipes, anything that produces a sound in Hauptwerk is considered to be a 'pipe' and so must be listed within the *Rank* table in order to function. To keep the custom organ definition format small and manageable, there is no need to list pipes or samples individually. Instead sample filenames must follow a particular format based on the note number from which they would normally sound. Thus only the sample folders need to be specified in the *Rank* table. Attributes prefixed *Samples_...* define the set of sample files and their properties.

The number of 'pipes' and the starting note number define the virtual pipes that will exist. A sample is loaded once per virtual pipe, even if the same samples is shared across a range of notes (i.e. if the interval between samples is greater than one). This is necessary because Hauptwerk performs off-line processing on each sample, depending on the pitch and other properties of the pipe using it, in order to give best possible real-time performance. Hence the contents of the *Rank* table define the virtual pipes, the samples to be used and also determine the amount of memory that will be used for a sample set. You should thus not create more virtual pipe objects (by way of *Rank* entries) than are actually needed, otherwise memory would be wasted.

Each entry in the *Rank* table will appear to the user as an entry on the rank routing options screen, allowing the user to enable/disable and select a particular memory format and audio routing for it.

A rank and its pipes can be either *percussive* ('one-shot') or *sustaining*. Samples for the former are simply played from start to end when the 'pipe' is triggered. The later must have at least one loop and a WAV cue point as a release marker (or the release portion can be in a separate sample file). Whilst the pipe is sounding the sample is played looped, and a phase-aligned cross-fade is performed to the section starting at the release marker when the pipe stops sounding.

Multiple release samples are also supported, by specifying 'short note' and 'medium note' release sample folders and associated maximum note times. When a note is played Hauptwerk will then choose the best release sample, depending on how long the virtual pipe had been sounding, giving improved realism of the virtual acoustic when playing fast.

A range of attributes exist on the *Rank* table to adjust the amplitude, pitch and harmonic content of the pipes within the rank, and the response of each to the wind supply model and tremulants. Values are specified for note number 36 (the lowest C on a 61-note manual) and 96 (the highest C on a 61-note manual), whether or not the rank actually includes pipes of those note numbers, and the values for each other note number that exists within the rank are projected linearly from those based on the note numbers of the pipes.

The relationship between stops and ranks

It is important to understand that there is always a many-to-many relationship between stops and ranks. Creating an entry in either or both of the *Stop* and *Rank* tables will not cause any pipes to be playable. You must also have at least one entry in the *StopRank* table, which links the other two together, i.e. connects the division's key action to the rank's pipes dependent upon the state of the stop switch.

Think of the *Stop* entry as defining the stop switch and the *Rank* entry as defining the set of pipes. Without the *StopRank* entry there is no connection between them. *StopRank* entries define the actions between the division notes/keys and pipes.

Because the *StopRank* table provides a many-to-many relationship, a rank can optionally be referenced by more than one stop (unification), and a stop can optionally reference more than one rank (e.g. multi-rank stops or stops that borrow bass pipes from another rank).

Each *StopRank* entry defines the range of division notes/keys that should be connected and the pitch of the action. Special types of action are also allowed, as with couplers, such as pizzicato, reiterators and theatre organ traps. The *ActionTypeCode*, *ActionEffectCode*, *PipeMIDINoteNum036_PizzOrReitPeriodMs*, *PipeMIDINoteNum096_PizzOrReitPeriodMs* fields make those possible. Note that it is better to use a coupler for a special complex type of action (e.g. pizzicato or reiterating) if it will be required for many stops, since there will be less overheads on the computer's processor than by duplicating it for a large number of *StopRank* entries.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Couplers

An entry in the *Coupler* table defines a key action between two divisions, which will be conditional upon the state of the coupler switch.

Most couplers found on a classical organ can be created simply by including an entry in the *Coupler* table with its coupler code set to the appropriate value. The coupler code defines the source and destination divisions, the coupling pitch, and the division in whose divisional combinations its state will be stored/recalled.

However, for special case, if you want the source division to be other than one of the standard seven divisions, then you must set the *OverrideSourceDivisionToSpecifiedDivisionCode* field. The division for combinations is always determined by the coupler code, and is not overridden by that setting. You may also override the destination division and couple the destination division's key action (the default) or keyboard.

Also, if you want to couple at a mutation pitch or create a coupler with a special type of action, such as pizzicato, reiterating or theatre organ trap, then you should use one of the 'custom coupler' values for the coupler code. The *CustomCoupler_...* fields may then be used to define the coupling pitch and coupling actions.

Note that a standard unison key action is also generated by default for each division unless a unison-off coupler is specified for that division.

Using Hauptwerk's native tremulant model

Hauptwerk's native tremulant model uses special 'tremulant waveform' samples to define the waveform (LFO) shapes used to modulate the amplitude, pitch and harmonic content of its pipes separately in real-time when a tremulant is active. This ensures that the tremulant-affected sound remains perfectly synchronized across all pipes attached to a given tremulant and that tremulants can be started and stopped realistically whilst notes are sounding.

Such tremulant waveform samples are stored in a special Hauptwerk-specific format and are supplied as part of sample sets, organized by rank(s) into folders, with waveforms included at intervals across the compass of the rank (if they are designed for compatibility with the Custom Organ Design Module).

To use Hauptwerk's native tremulant model in a custom organ definition you just need to include an entry in the *Tremulant* table, which represents the switch used to turn the tremulant on and off and defines the tremulant rate. The tremulant movement for all ranks attached to a given *Tremulant* object will be perfectly synchronized. The tremulant code determines the division in whose combinations its state will be stored and recalled.

Then specify the *Tremulant* object's tremulant code for *Trem_TremulantCode* in any given *Rank* entry to make the rank be affected by that tremulant. Since different ranks often respond differently to a given tremulant, the set of tremulant waveforms is specified separately for each rank, rather than for the *Tremulant* object. The *Trem_...* attributes of the *Rank* table define the set of tremulant waveform samples to use for a given rank, i.e. define the modulation shapes for it.

At most one tremulant can be attached to any given rank within the Custom Organ Design Module.

Using real tremulant-affected samples

There is also simple native support for using real tremulant-affected samples (if preferred to Hauptwerk's tremulant model) via the *StopRank* table. Held notes optionally do not re-trigger.

To implement them for a particular real pipe rank you would need two entries in the *Rank* table; one for the set of 'untremmed' samples and one for the 'tremmed'.

You then need a *Stop* entry (not a *Tremulant* entry) to define the switch that will be used to switch between the untremmed and tremmed ranks, i.e. the switch that the user will see as the virtual tremulant switch, even though it is actually an entry in the *Stop* table. (You do not need any *StopRank* entries for that stop directly.)

Finally, for each virtual stop that connects to the rank, the *StopRank* entry should specify the 'tremulant switch' *Stop* entry for the *StopCodeToSwitchToAlternateRank* parameter, and specify the untremmed rank for *RankID* and the tremmed rank *AlternateRankID*. The key action generated will then connect the keys to the untremmed rank when the 'tremulant switch' stop is off, but to the tremmed rank when it is on.

Held notes can optionally re-trigger when switching between the two, dependent upon *RetriggerNotesWhenSwitchingBetweenNormalAndAlternateRanks*.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Hauptwerk's swell box model

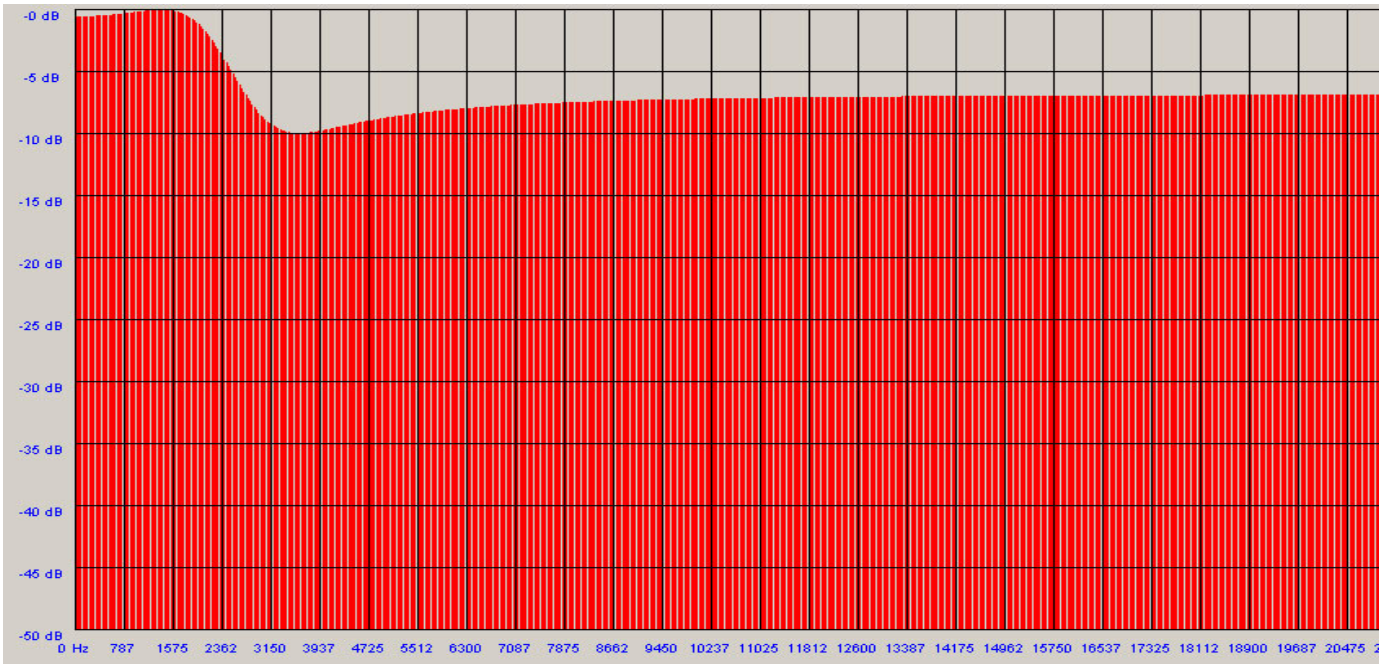
Hauptwerk models the effect of a swell box on each pipe individually by modulating its amplitude and applying a simple low-pass or high-pass filter to it. Because the filter is applied separately to each pipe in real-time, its frequency response is necessarily simple, so that the processor overheads are not excessive.

To implement a swell box, include an entry in the *Enclosure* table. The entry will then appear to the user on the relevant MIDI settings screen. A standard default shutter inertia and wind supply model can also be enabled for the virtual swell box.

To define which ranks are affected by a given swell box, set the *Encl_EnclosureCode* attribute for a *Rank* object to the enclosure's code. As with the native tremulant model, the effect of a swell box on the sound of the pipes is likely to vary from rank to rank and across the compass of the rank. The *Encl_FiltParamWhenClsd...* settings specify the shape of the filter's frequency response when the virtual box is closed and the *Encl_FiltParamWhenOpen...* for when it is open. Hauptwerk projects the values linearly in between for intermediate stages. As with other *Rank* settings, the responses are specified for a pipe with note number 36 (lowest C on a 61-note keyboard) and 96 (highest C on a 61-note keyboard), whether or not the rank actually contains pipes of those note numbers, and Hauptwerk projects the values for other notes from those.

The *...OverallAttnDb* settings define the amount of amplitude modulation. The *...MaxFreqHz* settings specify the frequency that will have the highest amplitude and *...MinFreqHz* the lowest. Thus to specify a low-pass filter, the *...MaxFreqHz* attribute would be set to a smaller value than its corresponding *...MinFreqHz* attribute. For a high-pass filter the opposite would be true. The *...ExtraAttnAtMinDb* settings specify by how the frequencies should be attenuated as a result of the filter, i.e. the extra attenuation at the minimum frequency response relative to that at the maximum (determined by *...OverallAttnDb* setting).

There are no hard and fast rules about the best settings since the frequency response of the enclosure filters has to be very simple (because they are applied to each pipe individually in real-time), so it is a case of fine-tuning the frequency response to give a result that sounds like a reasonable approximation for each rank. For example, the values for the St. Anne's, Moseley organ sample set were determined by making reference recordings of several pipes per octave for each rank with the swell box open and then with it closed. The difference in frequency responses were then plotted and an attempt was made to produce an approximate best-fit for the frequency response of Hauptwerk's swell box filters. The results were then further tweaked by



ear for each rank until they sounded reasonably similar to the reference recordings.

For a basic low-pass filter, which gives a reasonable result for any rank, and could thus be used as a starting point, you might wish to try:

- Maximum frequency when closed: 200 Hz
- Maximum frequency when open: 100 Hz
- Minimum frequency when closed: 1 kHz
- Minimum frequency when open: 8 kHz

The following image shows an example of the frequency response that would be obtained if the maximum frequency was set to 2 kHz, the minimum to 4 kHz and the extra attenuation at the minimum to 10 decibels.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Action noises, blower noise and sound effects

The custom organ definition format has special provision to make it simple to include action noises and sound effects.

To include such sounds, first list them in the *Rank* table. It is usually advisable to have a separate rank entry for each type of noise/effect so that the user has control over which to enable and to which speakers they should be routed.

Although a noise/effect might not have a well-defined pitch as such, it must still be associated with a note number and Hauptwerk must still be given a pitch for its sample(s). Provided that the pitch of the virtual pipes match those of the samples then no pitch-shifting will occur. For example, to include a single blower sample, you could simply use 060-C.wav for its filename, then set:

- *Samples_RankBasePitch64ftHarmNumIfAssumedTunedToConcertPitch* to *8* (meaning 8'),
- *Pitch_RankBaseOutputPitch64ftHarmonicNum* to *8,*
- *MIDINoteNumberOfFirstPipe* to *60,*
- *NumberOfPipes* to *1*

Because the sample and output pitches match no pitch-shifting would occur in that case.

Once the appropriate entry exists in the *Rank* table, you can attach it to any stop, coupler or tremulant using the following attributes on their tables:

- *PercussiveEngagingSoundEffect_RankID*
- *PercussiveEngagingSoundEffect_MIDINoteNumber*
- *PercussiveDisengagingSoundEffect_RankID*
- *PercussiveDisengagingSoundEffect_MIDINoteNumber*
- *SustainingSoundEffect_RankID*
- *SustainingSoundEffect_MIDINoteNumber*

For a blower, you would normally want to create a *Stop* entry for it so that its switch is accessible to the user and use the stop's *SustainingSoundEffect_...* attributes to attach the virtual 'pipe' (note number 60 in this example) to that stop. The sample will then sound as long as the stop is turned on. The stop could be made to default to the engaged state.

Theatre organ 'toy counter' effects can be implemented in the same way. By setting the *Stop*'s *SwitchIsLatching* setting to *N*, the stop will behave as a momentary button (piston), only staying on whilst it is held in by the user.

A tremulant 'chuffing' noise could be implemented similarly, this time just using the *SustainingSoundEffect_...* attributes for the *Tremulant* object so that it runs whilst the tremulant switch is on.

For 'clunks' as stops, couplers and tremulants engage, use the *PercussiveEngagingSoundEffect_...* attributes, and the *PercussiveDisengagingSoundEffect_...* attributes for sounds that should be triggered when they disengage.

If you have more than one noise/sound affect, e.g. a collection of stop action 'clunks' all listed as different notes in the same *Rank* entry, and want to attach a random 'clunk' to each stop, coupler and tremulant, simply set the relevant *...SoundEffect_MIDINoteNumber* to 0. If the setting is 0, Hauptwerk will choose a random 'pipe' (noise sample) from the rank.

Key action noise is implemented by listing one or more sets of key action 'clunks' as ranks in the *Rank* table. If you had separate 'key-on' clunks and separate 'key-off' clunks, then you would need two ranks; one for each. Then use the following attributes on the *Division* table to attach them to the division's keys:

- *PercussiveEngagingKeyActionNoise_RankID*
- *PercussiveEngagingKeyActionNoise_ChooseMIDINoteNumRandomly*
- *PercussiveDisengagingKeyActionNoise_RankID*
- *PercussiveDisengagingKeyActionNoise_ChooseMIDINoteNumRandomly*
- *SustainingKeyActionNoise_RankID*
- *SustainingKeyActionNoise_ChooseMIDINoteNumberRandomly*

If a *..._ChooseMIDINoteNumberRandomly* setting is set to *Y* then each key will be attached to a random 'pipe' (noise sample) from the rank, otherwise the key's MIDI note number will determine the 'pipe' to which it is attached.

For some examples of all of these things, see the *ExampleCustomOrgan5-StAnnes-WithBasicGUIAndNoises* example custom organ definition file.

You can alternatively implement any of these types or noise, or more complex cases, by creating additional entries in the *StopRank* table with the appropriate action types. However, the above settings are intended to make it as easy as possible to handle common requirements, avoiding the need for an inelegant number of entries in the *StopRank* table.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Samples and installation packages

Any organ definition is dependent upon the samples and other data it uses being installed upon the user's computer, and at the required version. This is particularly important for the Custom Organ Design Module, since it will often be used to combine ranks from various sample sets, and any organ definition files created with it may be distributed to other users who may or may not have all of those sample sets.

Hauptwerk thus considers sample set data (of all types except for the organ definition database itself, i.e. audio samples, tremulant waveform (LFO shape) samples, images and organ information files) to be contained within discrete *installation packages*, being conceptually bundles of sample set data which are installed and managed as units by Hauptwerk.

Each such installation package has a globally-unique identifier code which is allocated upon request to sample set developers from a central pool, a version, and supplier information, all defined in a file called the *package definition file* with filename extension *.InstallationPackageDefinition_Hauptwerk_xml*, located in the root of the folder containing the installation package data, termed the *installation package folder*.

Each organ definition references data (samples, images, etc.) via its installation package code, and lists the installation packages upon which it is dependent, together with the versions of those packages that are required.

When a user loads an organ definition file, Hauptwerk first checks that the required packages are installed and are at an appropriate version. If they are not, a message is given to the user, detailing the supplier of the package who should be contacted to obtain it (which may be subject to licensing and require purchasing). If all package prerequisites are met, Hauptwerk automatically constructs the full file paths to files according to the installation packages within which they are contained. File paths are always specified relative to the root of the installation package folder.

Hauptwerk handles installation and upgrade of installation packages natively, and any data within installation packages must never be edited manually except during the initial creation of a sample set (which is the subject of the separate guide: *Creating Sample Sets for Hauptwerk*).

If it is wished that the data within an installation package be edited, and the license from the original creator of the sample set allows it, then a new installation package should first be created, using a unique package identifier from the 'user' range (see below), and any references to the package should be updated accordingly in the organ or custom organ definition file. In this way, should the organ or custom organ definition file be shared with other users, Hauptwerk's installation package management system will continue to work properly, ensuring that the correct data are installed and that the original sample set is not broken for other users of the original sample set.

For your own personal use, please create packages with IDs in the ‘user’ range 800000 to 899999. If you wish to make a sample set available publicly, please contact Milan Digital Audio to be allocated a unique ID for you installation package (there is no charge for the service), and consult the Creating Sample Set for Hauptwerk guide for full details about how to format the package for Hauptwerk’s component installer.

Image files also may only exist in installation packages, and must be treated in the same way as samples with regard to their IDs and packaging for public distribution.

Installation package compatibility with the module

Not all sample set samples are necessarily compatible with the Custom Organ Design Module, since some additional requirements are placed upon them. For example, some sample sets may use layered samples, which are not currently supported by the module. Any compatible ranks of pipe samples will usually be listed in the package definition file in the *CustomOrganRank* table, and sets of tremulant waveform (LFO shape) samples in the *CustomOrganTremulantWaveformSet* table. Hauptwerk's XML file format is covered later with reference to the custom organ definition file format. The format is the same for both files, except that different tables are allowed and the file has different headers to identify its contents.

The compatibility requirements are covered fully in the separate Creating Sample Sets for Hauptwerk guide. They are summarized here for reference:

- All pipe sample files must have a names of the form *036-C.wav*, *037-C#.wav*, *038-D.wav*, ... (or with *.hbw* filename extensions) according to the MIDI note number and name of the keyboard key from which they would normally sound when played at unison pitch (i.e. with no explicit re-pitching). For unified (theatre) organs, the name would normally be set according to the note to which the sample would be attached when played at 8' pitch.
- Sustaining pipe sample files must either have the start of the (primary) release samples identified within the main pipe sample file by a marker (recommended in most cases for simplicity), or the release samples must be stored in separate files with the same names as the main attack/sustain sample but located in a separate folder.
- All main attack/sustain pipe sample files for a given rank must be contained within the same folder in the relevant installation package folder; one folder should be used for the attack/sustain samples for each rank. If the primary releases are stored in separate samples, then they too must all be located in a single folder for the rank, and it must be within the same installation package as the attack/sustain samples for that rank.
- If additional (multiple) release samples are used, then each set must be located in a separate single folder for the rank, and it must be within the same installation package as the attack/sustain samples for that rank.
- All pipe samples must either have their exact pitches detected and stored in the sampler chunks within the sample files (recommended), or be re-tuned perfectly to concert pitch and equal temperament. The pitch may not be

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

specified 'manually' in the organ database for a sample.

- Tremulant waveform samples must have names of the form: *060-C-TremulantPitchAndFundamentalAmplitudeWaveform.wav* and *060-C-TremulantThirdHarmonicAmplitudeWaveform.wav* (or with *.hbw* filename extensions) according to the MIDI note number of the keyboard key for the pipe which would normally sound when played at unison pitch and to which they are to be assigned.
- Tremulant waveform samples for a given rank must all be located within a single folder (although a rank's tremulant waveform samples can be assigned to more than one pipe rank by the user).
- Tremulant waveform samples must have their exact pitches in Hertz saved in the sampler chunk in the samples, pre-multiplied by 128 (since it is not possible to store a pitch value of less than about 8 Hz in a WAV file).
- All tremulant waveform samples must have at least 10 ms before the start of their loop and the loop must be at least 10 ms in length.
- Although not mandatory, it is recommended that an additional release sample marker is included in the release sample (region) for any ambient ('wet') samples at the point at which the sound starts to decay. This allows Hauptwerk's automatic reverberation tail truncation functionality to work optimally, allowing ambient samples to be played either 'wet' or 'dry' and combined with ranks from different organs, or played at non-unison pitches with the release samples being adjusted automatically to an appropriate length for the playback pitch.

These requirements are made so that the user of the custom organ design module does not need to specify filenames and parameters for every pipe or sample.

Determining the sample settings for pipe and tremulant waveform samples for the Rank table

If you are using samples from a third-party sample set in your custom organ definition, you can usually find the values you need for the *Samples_...* and most of the *Trem_...* settings on the *Rank* table by looking in the installation package definition file at the contents of the *CustomOrganRank* and *CustomOrganTremulantWaveform* set tables respectively.

For example, to see the settings you would need to use in the *Rank* table if you wanted to use some samples from the St. Anne's, Moseley sample set, open the *PackageID000010.InstallationPackageDefinition_Hauptwerk_xml* file from the *000010* folder in the *OrganInstallationPackages* folder in the *HauptwerkSampleSetsAndComponents* folder using a text or XML editor.

Wet and dry samples and the reverb tail truncation functionality

In general, pipe samples recorded from different organs can only be combined effectively if they were recorded 'dry' (without room ambience or reverberation), since the differences in room acoustics would otherwise be obvious when they were played together.

However, Hauptwerk also includes a feature whereby release samples containing room ambience can automatically be trimmed and shaped to model the decay of pipes recorded 'dry' (without room ambience). Parameters can be set for each rank in the custom organ definition database controlling its operation and the user is able to override those parameters when adjusting the voicing of a layer. Thus the user is able to select to load all or part of an organ with its full release samples or with those samples truncated to model dry recordings.

A shaped fade-out is used, of a length determined by the layer or voicing parameters, and adjusted automatically for the pipe frequency (since it takes longer for a speaking bass pipe to fall silent than a treble pipe). The fade is designed so that, as far as possible, the initial transients of the release are preserved, whilst the decay is truncated to model the shape of the natural decay of a pipe recorded dry for the appropriate pitch.

This makes it feasible to combine semi-dry ranks from various sample sets fairly effectively, significantly reducing the effects of any differences in acoustics, which would otherwise render their combination impossible. Within the Custom Organ Design Module, it is also possible to assign ranks to be played at pitches other than that at which the samples were recorded without the release samples becoming unnaturally long or short.

Of course simply shaping the releases does not provide a perfect model of real dry recordings, especially since the room acoustics affect the frequency response of the samples themselves, and the shapes of their attacks.

Filenames and paths

Note that Hauptwerk is a multi-platform application, and that filenames are handled slightly differently on different platforms. On UNIX-based systems, such as macOS, folder levels are separated by a '/' character, whereas on Windows systems, the separator is a '\' character. Either character may be used in path references in the organ definition database and elsewhere in Hauptwerk; Hauptwerk will perform the translation automatically for the system on which it is running.

However, filenames and paths may be *case-sensitive* on UNIX-based systems, such as macOS, while Windows systems are not. To ensure compatibility across all supported Hauptwerk platforms, *the case of filename and path references in the organ and custom organ definition databases must match that of the files and folders exactly*. Hauptwerk does not perform automatic translation in this case.

Also, where the format of file and folder names or file extensions is specified in subsequent sections, *the case must match that specified exactly*, otherwise an organ might not function on macOS.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Due to the different ways in which the various file systems work you should *avoid using any characters in filenames or paths which are not present in the standard 7-bit American ANSI character set*. For simplicity, it is strongly recommended that only the following characters be used: '*A*'-'*Z*', '*a*'-'*z*', '*0*'-'*9*', '-', '_'. The space character is generally best avoided, and the '.' character should only be used in file extensions as specified for each file type. Regional characters, such as those with umlauts or cedillas might prevent an organ from loading on computers in some parts of the world.

Globally-unique organ identifiers and making organ definitions available to others

Each organ is identified internally within Hauptwerk by a six digit globally-unique identifier, the organ ID. This identifier is used by Hauptwerk to associate all organs-specific settings and cached sample set data with the organ.

If two organ definition files with the same organ ID were ever to be installed on the sample computer, then the settings and sample data from one would overwrite those of the other, with disastrous results for the user. To avoid this possibility, the organ ID is allocated upon request to Milan Digital Audio from a central list.

However, a special exception is made for the Custom Organ Design Module so that its users are able to create custom specifications for their own personal use with the minimum of fuss. The unique organ ID for any custom organ must lie within the special reserved range of 800000 to 899999.

If you receive a custom organ definition file from another user then you must first open it and ensure that the organ ID within the range 800000 to 899999 is unique amongst those that are present on your computer.

If you use the Custom Organ Design Module to generate a native Hauptwerk organ definition file that you wish to supply publicly as part of a sample set, then you need to open the generated native organ definition file (with an extension *.Organ_Hauptwerk_xml*) and change the organ ID to a unique value that Milan Digital Audio has allocated to you for it. Please simply contact us to request allocation of such a value if you need it. There is no charge for the service. No other edits are necessary to a generated organ definition file to make it suitable for public distribution as part of a sample set.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

The custom organ definition file XML format

The custom organ definition files must have filename extensions of *.CustomOrgan_Hauptwerk_xml*, and be located in the *CustomOrganDefinitions* folder within the *HauptwerkUserData* folder, the location of which is determined during the original installation of Hauptwerk.

They may be edited directly in any text editor, such as Windows' Notepad, or UltraEdit (the latter is recommended and is available for a very reasonable cost from *http://www.ultraedit.com/*), or using a dedicated XML editor program, such as the excellent freeware XML Marker (*http://www.symbolclick.com/*).

XML is a simple format for storing and exchanging structured data. Data are enclosed within 'tags', which may also contain further 'sub-tags'. An enclosing pair of tags take the following form:

```
<tag>data</tag>
```

Here *tag* is the name of the tag, and *data* is a set of data enclosed by the tags. A tag may also have attributes. In the following example *ObjectList* is the tag, *ObjectType* is an attribute of the *ObjectList* tag, *CustomDisplayPage* is its value, and the ellipsis shows the location of the data set enclosed by the *ObjectList* tag:

```
<ObjectList ObjectType="CustomDisplayPage">...</ObjectList>
```

The following example shows how tags are nested within other tags, with the attributes (table fields) belonging to the object (table row) that encloses them, and the object (table row) belonging to the object list (table) that encloses them:

```
<ObjectList ObjectType="customdisplaypage">
	<customdisplaypage>
		<DisplayPageID>1</DisplayPageID>
		<Name>Console</Name>
		<BackgroundImageFilename>HauptwerkStandardImages/DarkCherry.png</BackgroundImageFilename>
		<BackgroundImageFileInstallationPackageID>1</BackgroundImageFileInstallationPackageID>
	</customdisplaypage>
	<customdisplaypage>
		<DisplayPageID>2</DisplayPageID>
		<Name>Left Jamb</Name>
		<BackgroundImageFilename>HauptwerkStandardImages/DarkCherry.png</BackgroundImageFilename>
		<BackgroundImageFileInstallationPackageID>1</BackgroundImageFileInstallationPackageID>
	</customdisplaypage>
	<customdisplaypage>
		<DisplayPageID>3</DisplayPageID>
		<Name>Right Jamb</Name>
		<BackgroundImageFilename>HauptwerkStandardImages/DarkCherry.png</BackgroundImageFilename>
		<BackgroundImageFileInstallationPackageID>1</BackgroundImageFileInstallationPackageID>
	</customdisplaypage>
</ObjectList>
```

The custom organ definition file must have the following opening tags immediately at the start of the file:

```
<?xml version="1.0" encoding="UTF-8"?>
<Hauptwerk FileFormat="CustomOrgan" FileFormatVersion="6.00">
```

… with the corresponding closing tag immediately at the end:

```
</Hauptwerk>
```

All of the data lying between are organized into object lists, objects and object attributes as illustrated in the example above. These object lists, objects and object attributes are generally referred to as *tables*, *table rows* and *table fields* in the remainder of this document, since they can be thought of as being a representation of two-dimensional tables of data. The structure of each table is described later in this guide.

Note that, for attributes which are not mandatory, the absence of a value should be denoted using the following format:

```
<ObjectList ObjectType="division">
	<division>
		...
		<WindModel_WindchestPressureDropPctAtMaxLoad></WindModel_WindchestPressureDropPctAtMaxLoad>
		...
	</division>
</ObjectList>
```

Indentation and other 'white space' is ignored in the file unless it is within a quoted attribute value or field value. Certain 'special' characters, which are used to define the structure of an XML file must be represented by 'escape' sequences when they are used in data or attribute names in XML, as in HTML:

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Character	Escape sequence
'<' (less than sign)	<
'>' (greater than sign)	>
'&' (ampersand)	&
'"' (double-quote)	"

Other characters can be embedded as normal, provided they are encoded correctly within the specified XML character set (for example UTF-8).

As an example, the field value 'GT & PED PISTONS COUPLED' would be represented as follows:

```
<Name>GT &amp; PED PISTONS COUPLED</Name>
```

The recommended character encoding (the actual byte values used to store each character) in the file is UTF-8, a 'universal' character encoding which can represent almost all characters from around the world. In general, this is only relevant if any characters other than the standard American (ANSI) characters are used; characters with umlauts, for example. The UltraEdit text edit allows the encoding of a text file to set to UTF-8, so that any characters can be used safely. However, Windows' Notepad is less easy to work with for UTF-8 text files.

Note that dedicated XML editors, such as the freeware XML Marker, can be used to handle character encodings, escape sequences and tags etc. natively. The single biggest benefit of XML as a format is that it is a widely-accepted standard, and there are many existing third-party tools that can read, write and manipulate it.

The sample set data cache

When Hauptwerk first loads an organ, all signal processing that does not need to happen in real-time is performed on the audio samples, and the resulting samples are saved into its internal 'sample set data cache'. Image files are also stored within the cache to allow faster subsequent loading. The files within the sample set data cache cannot be edited.

Certain changes to Hauptwerk's user configuration (especially audio routing) require the samples cached for a given organ to be regenerated. Hauptwerk normally handles such regeneration automatically. However, when the design of an organ is adjusted, either within a full organ definition file or a custom organ definition file, it may or may not be necessary for the sample set data cache to be regenerated, depending on the nature of the change. For example, the addition of a pipe rank or a change to an image file would require the sample cache to be regenerated to reflect the change, but the addition of a coupler would not.

By default, Hauptwerk always regenerates the sample set data cache when a custom organ definition file is loaded, since it has no means to know the nature of any changes in its design that may have occurred. Since regenerating the cache is rather slow, an additional option is available when loading the file which allows cache regeneration to be skipped. This option must only be chosen when it is absolutely certain that no changes have been made to the ranks (or usually image files) listed in the custom organ definition. However, a further option allows images to be re-loaded from their source (non-cached) files, even when the sample data are loaded from the cache. This allows for fast testing of image and screen layout changes.

Organ-specific configuration, voicing and combinations

Similarly, the organ-specific configuration and user voicing stored for the organ might no longer be valid following a change to the design of a custom organ. By default, any such existing configuration is cleared entirely when a custom organ definition file is loaded (unlike a full organ definition file).

However, further options are available when loading the custom organ to override this behavior, retaining various parts of the organ-specific configuration and user voicing, should it be certain that no changes have been made which affect the relevant parts of the configuration.

When loading a custom organ using the *Design tools* | *Load custom organ* … menu option the *Load Organ Design Options* screen is always shown to you for this purpose, allowing you to tell Hauptwerk when changes have been made in the organ definition that would necessitate resetting its organ-specific voicing or configuration.

The Custom Organ Design Module is designed to preserve existing user voicing if a rank is added or removed, provided that the IDs of the remaining ranks are not changed. It also preserves existing organ-specific MIDI configuration for virtual objects when virtual keyboards, stops, couplers, tremulants, pistons and swell boxes are added or removed, again providing that those that remain are not assigned new IDs (since it identifies each type of object by its ID). General and divisional combinations, as well as combination stepper frames, stored in combination files also remain valid in these circumstances.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.

Tables in the custom organ definition database

For full details of the tables and attributes, including full descriptions, please use the '*Design tools* | *View Custom Organ Design Module … format documentation*' function within Hauptwerk itself. The documentation that function shows should be read in concert with this user guide, and forms an important, integral part of the documentation for the Custom Organ Design Module.

Importing and exporting the custom organ definition database to SQLite

Since a database for a complex organ can be quite large, it is often not easily possible to edit the definition files directly using just a text editor. XML manipulation tools can be used instead, but most are fairly limited. Hence Hauptwerk has special functionality to import and export organ definition files to/from a third-part cross-platform open database format, called SQLite (version 3).

Use the *Design tools* | *Export custom organ to database …* menu function in Hauptwerk to select an organ definition file to export and to choose a name for the SQLite database file to be exported. Hauptwerk will then load the organ data from the definition file and generate a SQLite database of the specified name. Note that any existing database of the same name will be overwritten without any warning, so caution should be exercised. Note also that any organ currently loaded within Hauptwerk will be unloaded prior to the export being performed.

With an organ definition exported, its data can be edited as required using Navicat for SQLite (see next section), or other third-party SQLite front-end editor tools. Each SQLite organ database is defined entirely by a single file within the *CustomOrganDatabase* sub-folder of your Hauptwerk installation's *HauptwerkUserData* folder.

Upon completion of the required edits, use the *Design tools* | *Import custom organ from database …* function to select the database and choose a filename for the resulting organ definition file. Hauptwerk will then read the SQLite database, import its data to Hauptwerk's custom organ definition (XML) file format, saving the organ definition file with the specified name, and load the resulting file as normal for a custom organ. Note that any organ definition file of the selected name will be overwritten without any warning, so this function too should be used with caution. Note also that exporting and importing may be quite slow for large databases.

Browsing and editing data using Navicat for SQLite

Various third-party GUI front-end editors are available for SQLite, including some freeware ones. We recommend Navicat for SQLite (http://www.navicat.com/), which is very powerful and has full and convenient support for all of the features that Hauptwerk's exported databases require, including foreign keys, default column values, integrity check constraints, stronger data-typing, scripting, and sub-queries (which weren't all available in some of the other editors we tried).

To view and edit an exported Hauptwerk database within Navicat for SQLite, create and open a connection to the corresponding database file within the *CustomOrganDatabase* sub-folder of your *HauptwerkUserData* folder. You will then be able to browse and edit the data in the tables, or use the SQL language to manipulate the data in more-complex ways.

Databases are created, deleted and populated by Hauptwerk using the export/import menu options on the *Design tools* menu as covered in the preceding section. Note that the structure of Hauptwerk's databases should never be edited; the tables, columns, indexes and so forth define the structure, and are determined by Hauptwerk. Changing the structure in any way is likely to prevent Hauptwerk from being able to load the database. It is only the data within the tables that should be edited. Note also that you should never change any of the data in the tables named with *__ValidFixedCodes_*... prefixes; the data they contain are fixed by Hauptwerk and exported for reference (and integrity enforcement via for foreign keys) only.

Impulse response reverbs

If you wish to make or supply impulse response reverbs in Hauptwerk format, then please consult the '*Design tools* | *View … reverb format documentation*' menu function, which includes a section describing the format of impulse response reverbs in detail.

© Copyright Milan Digital Audio LLC. 2001-2022 All Rights Reserved.