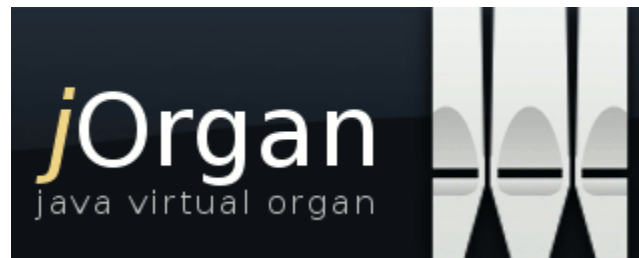


# *A Beginner's Guide to*



**Volume 3**

**Bill Skees**

**with contributions from the jOrgan Community**

4/3/2009

Rev. 9/30/2010

## **Dedication**

To all of those whose imagination, creativity, and tireless efforts have contributed to the state of the art in Virtual Organ technology—most especially Sven Meier who developed and continues to improve jOrgan.

## **Acknowledgments**

I am grateful for the many kindnesses of all the members of the jOrgan community as displayed in their willingness to share the fruits of their creativity with the Virtual Organ community at large and in particular their willingness to provide help and advice in the development of this manual.

## **Editorial Usage**

This manual follows the traditional use of the words “he, him, his, etc.” when the context implies the impersonal, as in “programmer, user, beginner, etc.”

## **Caution**

It is the nature of the Internet to be continually changing. Consequently the reader of this manual may find that a given website reference is no longer valid or that the contents of some sites may have changed from what is presented herein. While we have done our best to check all our material for accuracy as of the latest revision date of this document, contributors to this manual cannot be responsible for any differences that the reader may encounter on actual websites or for any websites that are no longer in existence.

## **Computer Compatibility**

As of this writing all of the software discussed in this document is known to be operational on PC computers under Windows XP and Vista operating systems unless otherwise stated. Work is indeed underway on making the software fully Mac compatible but this capability has not yet been realized. For these reasons please consider all text herein to apply only to PC Windows XP and Vista unless specifically indicated otherwise.

## **Legal and Ethical Issues**

This document is intended to assist in making the reader’s experience with jOrgan intellectually and aesthetically enjoyable. jOrgan is designed to be compatible with a vast spectrum of software and hardware products. Our assumption in providing you with information about interfacing jOrgan to other products is that you, the reader, are legally and ethically entitled to the use of such products and that you will use responsibly whatever you learn about jOrgan from this manual.

# Contents

<b>Introduction</b> .....	<b>iv</b>
Why This Manual Was Written and Who Needs It .....	iv
Organization of This Manual .....	iv
Technical References .....	iv
People, Sources, Websites and Other References .....	iv
<b>16. jOrgan Beginning Disposition Designers Corner</b> .....	<b>16.1</b>
Introduction .....	16.1
Order of Divisions On Draw-Knob Dispositions .....	16.1
Sound Source Planning—Sample Pitch Detection .....	16.1
Virtual MIDI Channels (Virtual MIDI Cables) .....	16.2
<b>17. jOrgan Beginning Disposition Developers Corner</b> .....	<b>17.1</b>
Introduction .....	17.1
Architecture of a jOrgan Disposition .....	17.1
Crescendo Pedals .....	17.3
Hauptwerk-Like Couplers in jOrgan—Swell Melody and Pedal Bass .....	17.3
Recorder Element—Using .....	17.3
Soundfont Importing .....	17.4
Transpose Coupler Implementation .....	17.4
Unison Off Implementation .....	17.5
Voicing Command Implementation .....	17.6
<b>18. jOrgan Beginning Developers MIDI Language Corner</b> .....	<b>18.1</b>
Introduction .....	18.1
Seeing (And Modifying) MIDI—What MIDI Information Looks Like .....	18.2
<b>19. jOrgan Beginners Introduction to Extensions</b> .....	<b>19.1</b>
Introduction .....	19.1
Creative Soundcards Feature .....	19.2
FluidSynth Sampler Feature .....	19.2
Import Elements Feature .....	19.2
Virtual Keyboard Feature .....	19.3
LAN Devices Feature .....	19.3
LinuxSampler Configure Feature .....	19.3
Memory Feature .....	19.4
Merge MIDI Devices Feature .....	19.4
Record Performances Feature .....	19.5
SAMS-Driven Stop Tabs Feature .....	19.7
Soundfonts Usage Feature .....	19.7
Execute Arbitrary System Commands Feature (Executor) .....	19.7
MIDI Tools and Utilities Feature .....	19.8

# Introduction

## Why This Manual Was Written and Who Needs It

This manual is a from-zero start up manual for people who are unfamiliar with jOrgan but who want to build or use a jOrgan Virtual Organ, whatever that turns out to be. No previous knowledge of either real organs or virtual organs is required. One need not be a musician or even a computer programmer. Simply being able to operate a computer (particularly a PC) and having an interest in making musical sounds is enough. Be prepared to enjoy yourself!

## Organization of This Manual

Chapter 1 discusses the different roles that jOrgan fulfills in the world of music and creative technology in the arts. Chapter 2 provides illustrations of increasingly more sophisticated implementations and applications of jOrgan technology. Chapter 3 and later chapters present step by step guidance on how to install and operate examples of the kinds of jOrgan applications that were introduced in chapters 1 and 2, and how to build your own jOrgan consoles.

The Appendices are intended to serve as references for the earlier chapters, containing more detailed technical information as well as suggestions for other sources of jOrgan information.

The Beginners Manual has been divided into four volumes. Volume 1 contains chapters 1 through 7. Volume 2 is chapters 8 through 15, Volume 3 is chapters 9-16, and Volume 4 is appendices and glossary.

## Technical References

The most authoritative source of information about the inner workings of jOrgan itself is provided on Sven Meier's site: <http://jorgan.sourceforge.net/introduction>.

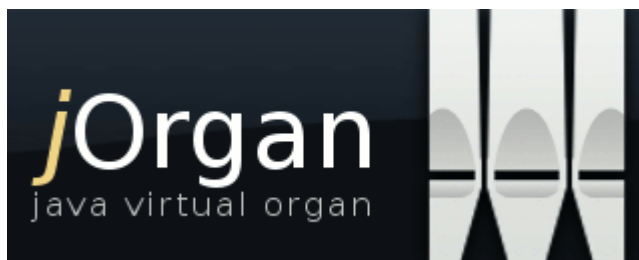
## People, Sources, Websites and Other References

Many people have contributed generously and enthusiastically to the material presented in this Manual, most of whom I know only through their membership in the jOrgan community discussion group at [jorgan-user@lists.sourceforge.net](mailto:jorgan-user@lists.sourceforge.net). This group includes, but is not limited to, Tony (Tony Meakin, France), Mark B. (Dr. Mark Bugeja MD, Malta, Europe), Mark, Bill B., Andy, Rick (Rick Whatson, Australia), Roy Radford, Lynn Walls, Bernd, GrahamG, John, Jacques, Bill Burd, Dan Dietzer, Lynn Walls, Joseph Hardy, John Reimer, Pete, Matt Pestle, Jim Ketcheson, Marco Francesco, Paul Kealy, ...

(more to be inserted)

Many people have made the contents of their websites available either to me directly, or to the jOrgan community, and I wish to thank them on behalf of the jOrgan community. These include Kent Allman, Paul Stratman, Bernd Casper, Graham Goode, Bruce Miles, ...

(more to be inserted)



## 16. jOrgan Beginning Disposition Designers Corner

### Introduction

This is a compendium of jOrgan forum topics that would be of interest to jOrgan Disposition designers—those people who decide what capabilities and characteristics their jOrgan dispositions will provide to the performer). Because much of this material is beyond what would ordinarily be encountered by the absolute Beginner I have not tried to validate or verify these entries. Please let me know if you find these entries a) unworkable, or b) inappropriate in any way for the Beginners Guide.

Where possible dates have been added to attributions to help you judge whether contributions remain relevant.

*Caveat: some of the material in this chapter is beyond my level of expertise so I may have unknowingly made mistakes in transcription or interpretation; also I may have unwittingly made typographical errors.*

### Order of Divisions On Draw-Knob Dispositions

*Question:* Why do jOrgan dispositions (such as ACO\_4.2\_3P-104\_FS, an American organ!) have an order such as: (left to right) Pedal, Choir, Great, and Swell when the order specified by the American Guild of Organists (AGO) is: Left jamb: Pedal and Swell and Right jamb: Solo, Great, Choir, Positiv?

*Answer:* The arrangement on the Main console is Pedal, Choir, Great, Swell because my disposition's monitor is on the left side of the keyboards and the divisions are arranged in that way—to mirror the positions of the keyboards, bottom to top, left to right. [Paul S. 8/4/2010]

### Sound Source Planning—Sample Pitch Detection

*Question:* Is there any freeware tool that I can use to accurately detect the pitch of a sample in Hz?

*Answer:* I use GTune (which is a VST plug-in) <http://www.gvst.co.uk/gtune.htm>. Do not use the one with the fancy skin. Basic version works better and gives you cents + and -, which makes adjustments with soundfonts so much easier. [Paul S.]

*Answer:* GTune located on website <http://www.gvst.co.uk/index.htm>. It is a VST plug-in so you will also need a VST host such as the one on <http://www.hermannseib.com/english/vsthost.htm>. [Dan 8/9/2010]

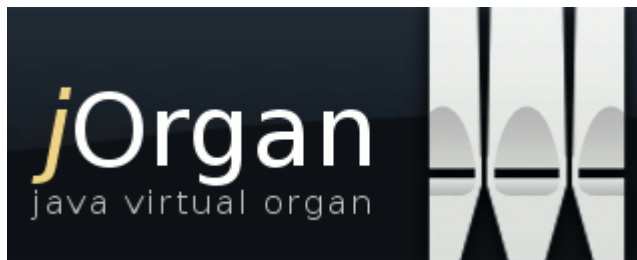
## **Virtual MIDI Channels (Virtual MIDI Cables)**

*Question:* Which Virtual MIDI Channel/Cable applications work with jOrgan?

*Answer:* (LoopBe, Maple & LoopBack) There is a known problem with jOrgan (java actually) hanging when attempting to close a loopBe virtual device. The problem has been reported to the loopBe developer but no fix has been announced. There are two other such tools: Marble Sound Maple Virtual MIDI Cable and Hubi's (Hubert Winkler's) MIDI LoopBack Device. [Lynn]

*Answer:* (MIDI Over LAN CP) the MusicLab product Midi Over Lan CP works with all Windows platforms and Mac. It can be used to midi-network computers, and has a loop back capability. It is not a free product, but of all I tested I found it the most stable [http://www.musiclab.com/products/rpl\\_info.htm](http://www.musiclab.com/products/rpl_info.htm). [Marc-Paul]

*Answer:* (MIDI Yoke) Midi Yoke installs and works with Win 7 64-bit. You have to disable UAC and install as Administrator. If Midi Ox can't see the Midi Yoke ports then you need to run the install again and reboot. All 64-bit programs will not see Midi Yoke, so you have to make sure you're using jOrgan in 32-bit Java. [Graham G.]



## 17. jOrgan Beginning Disposition Developers Corner

### Introduction

This is a compendium of jOrgan forum topics that would be of interest to jOrgan Disposition developers (individuals who actually build the elements of a jOrgan disposition). Because much of this material is beyond what would ordinarily be encountered by the absolute Beginner I have not tried to validate or verify these entries. Please let me know if you find these entries a) unworkable, or b) inappropriate in any way for the Beginners Guide.

Where possible dates have been added to attributions to help you judge whether contributions remain relevant.

*Caveat: some of the material in this chapter is beyond my level of expertise so I may have unknowingly made mistakes in transcription or interpretation; also I may have unwittingly made typographical errors.*

### Architecture of a jOrgan Disposition

Anyone who aspires to develop his own jOrgan disposition needs a general notion of how the various parts of a disposition work together. Lynn Walls has graciously provided the following summary:

A jOrgan disposition may be visualized as a network of “elements”—each having a filtering or transforming or controlling function—which are chained together in a one-to-many and many-to-one fashion via “references to” and “references from” linkages. What these references do is direct the flow of the MIDI message and other control information among the networked elements. Each element performs its defined function using information from its own “properties” values as well as information provided by its upstream references. Typically, an element will perform its function on incoming MIDI message data and render a result that consists of one or more of these:

- a copy of the incoming message
- a transformed version of the incoming message
- additional messages
- a blockage of the incoming message

that are passed on to the next elements in the reference chain.

At the beginning of the reference chains are the elements which collect triggering actions by humans—Keyboard, Activator, ContinuousFilter, etc. These latter are jOrgan elements which INITIATE a midi message flow.

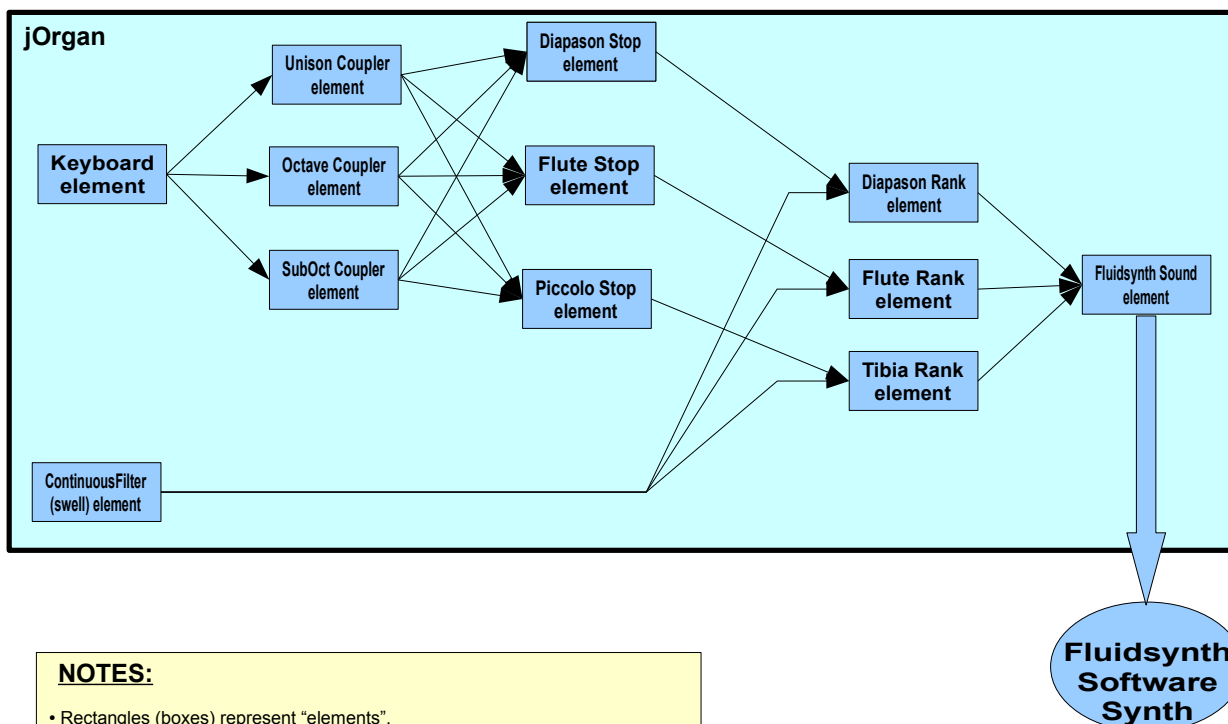
Some elements, such as SwitchFilter and Stop, are interposed in the message chain and block or allow or transform the messages that they receive from their upstream “reference from” elements before sending (or not sending) those same or transformed version of those messages to any “referenced to” elements downstream.

At the end of the chain are the Sound elements, which receive the copied, duplicated, transformed or internally generated midi messages and send them into the sound rendering interface (Fluidsynth, GSO, other software sampler/synth, Creative Sound Blaster, etc.).

For example, the Keyboard element captures a Note-ON midi message from an external MIDI data stream source. It references a Unison Coupler and an Octave Coupler. If the Unison Coupler is OFF, the Note-On message is passed on without alteration. If the Unison Coupler is ON, the Note-On message is blocked from further downstream flow. Now in parallel, the Octave Coupler is seeing the SAME incoming MIDI Note-On message. If the Octave Coupler is ON, the Note-ON data2 value is incremented by 12 and the resulting Note-ON message is re-transmitted to the downstream references with the data2 value set an octave higher than the original value from the Keyboard. If the Octave Coupler is OFF, the Note-On message is blocked from further downstream flow.

In this example, both the Unison Coupler and the Octave coupler receive a copy of the Note-On midi message from the Keyboard, and both may or may not send on a version of that Note-ON message to their downstream references, which would typically be a group of Stop elements representing the ranks that could be playable from the Keyboard. The function of those Stop elements would be to pass the incoming MIDI data stream to the next element or block it. That next element would typically be a specific Rank element. The purpose of the Rank element would be to set up a MIDI channel in the referenced Sound element and transform any incoming Note-On messages to that specific channel number transforming, if necessary, the data1 (pitch) and data2 (velocity) values as defined in the Message specifications for that Rank. Of course, the Rank element could add even more messages to the MIDI data stream if there were more than one “Note Played” message defined in its list of Messages.

Each Rank element would then reference a downstream Sound element. The Sound element is typically the element at the end of the information chain. The Sound element connects jOrgan to the outside world of sound. The Sound element has no “reference to” references because there is no more information flow beyond the Sound element within jOrgan. The Sound element sets up the interface to the sound source device or software, and passes the midi message that that it receives into this sound rendering device. This is really jOrgan’s point of hand-off. Even Fluidsynth, which appears to be integrated into jOrgan is really not a part of jOrgan. jOrgan’s Sound element simply hands off the MIDI message data to the internally packaged Fluidsynth in much the same way that it would hand off that data to an external program like GSO or a hardware sound source device like a Creative Sound Blaster card synth (see diagram). [Lynn Walls 9/16/2010]



#### NOTES:

- Rectangles (boxes) represent "elements".
- Arrows represent "references".
- Arrows that point TOWARD an element are a "reference to" that element.
- Arrows that point AWAY FROM an element are a "reference from" that element.
- Visualize the MIDI message data as flowing along the arrows.

## Crescendo Pedals Programming

*Question:* How do you program a crescendo pedal in jOrgan?

*Answer:* A Crescendo is made up of Activator elements that are referenced by a Regulator element. So, create your first Activator - name it Crescendo-1, leave it blank so that your Cresendo has a silent base, then create the second Activator, name it Crescendo-2, and this time reference the Stops that will be activated as the first step of the Cresendo etc., until you have eight or ten steps. Then create a Regulator, reference all the activators that you have just created and configured, then add it to the console and select the Expression Pedal from the skin. [Graham G. 9/10/2010]

## Hauptwerk-Like Couplers in jOrgan—Swell Melody and Pedal Bass

*Question:* Is it possible to make a swell melody feature and pedalbass (like in the Sample Hauptwerk set)?

*Answer:* Make a coupler. Select stops as you would for any other coupler and reference it to the desired keyboard. Under properties, "Action" select "highest" for melody or "lowest" for bass. [Paul S. 8/7/2010]

## Recorder Element—Using

*Question:* How do I get Recorder to run in the disposition I've created?

*Answer:* When you add a Recorder element you must cause it to reference your Keyboard elements and your Console element. If you don't do this, the recorder will not know which midi data to record, and its window will not have any midi data sources shown. [Lynn 7/27/2010]

## Soundfont Importing

*Question:* Do both Fluidsynth and Creative Sound load the soundfont into the assigned-base bank?

*Answer:* jOrgan doesn't take the preset's bank into account. [Sven 8/10/2010]

## Transpose Coupler Implementation

*Question:* Is it possible to have a transpose function, where the note on and off messages are translated to other messages as you play?

*Answer:* You may want to construct a coupler transposer. For each keyboard you will make 13 couplers. Twelve will be labeled and will transpose GT\_-5, GT\_-4, GT\_-3, GT\_-2, GT\_-1, GT\_0, GT\_+1 . . . GT\_+6. The thirteenth will be labeled Great\_Master (or whatever the keyboard is named.)

The master coupler will have all the keyboard's tops referenced to it, and it will be "Active = True." The twelve transposing couplers will be referenced to their respective keyboard. (Using a master coupler reduces the number of references.)

Twelve activators reference the couplers. (GT\_-5, SW\_-5, PD\_-5 would all be referenced to the -5 activator, etc.) Put the twelve activators into a regulator with + and - incrementers. This kind of coupler transposer works with any soundsource. It preserves scaling and mixture breaks. [Paul S. 8/12/2010]

## Unison Off Implementation

*Question:* I'm having a problem setting up a Unison Off Coupler on the Great. The Coupler element is set as follows:

Action: Inverse

Duration: -1

Transpose: 0

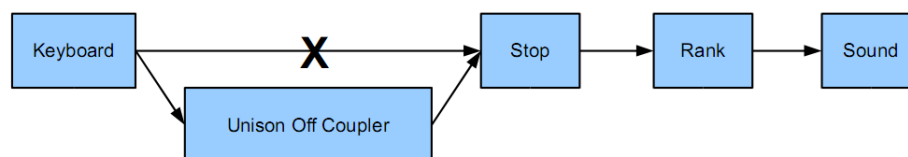
Ref To: Stops on the Great

Ref From: Great Console and Other Couplers that affect the Great

When I active the Unison Off Coupler, the playing pitches are not interrupted, they just keep playing until I release the notes on the Great.

*Answer:* Typically this behavior indicates that the Stops are referenced via the Keyboard element as well as via the Unison Off Coupler. When using the Unison Off coupler you also must remove any references from those stops to the keyboard element referenced by the coupler. The coupler will now send (or not send) those MIDI messages to the keyboard element. [Ted]

Visualize the relationship of the jOrgan ELEMENTS (boxes) and their REFERENCES (arrow lines) as shown:



Once the Unison Off Coupler ELEMENT has been inserted into the Reference chain between the Keyboard element and the Stop element, the original Reference from the Keyboard ELEMENT to the associated Stop ELEMENTs must be removed (“X” in graphic). Otherwise, input from the Keyboard element will continue to BYPASS the Unison Off Coupler (which may be thought of as an on/off switch in the Reference chain). [Lynn]

*Question:* Should the “Unison (meaning=?) Off” coupler only reference 8 foot stops? And should its Action in jOrgan be straight or inverse?

*Answer:* No. Unison means: “Play at the pitch specified by the Stop -- NOT play at 8’ pitch.” When you turn the Unison Off coupler ON you want all Stops to cease playing at their designated pitch -- not just the 8’ Stops. Presumably you will have one or more OTHER pitch-shifting couplers (SUB, OCT, “Blackpool”) turned ON. In the absence of other couplers being ON, turning ON the Unison Off coupler should yield SILENCE from ALL affected Stops.

Inverse -- When in OFF position, the switch should be ON (passing the Reference through). [Lynn]

The transpose value of the Unison Off stop should be zero, as opposed to +12 for octave, -12 for sub etc. This will cause the coupled ranks to play at their natural pitch. [Roy, Lynn 8/16/2010]

## Voicing Command Implementation

*Question:* The Rodgers 340 has “voicing tabs.” For example the voicing tab for the Tibia family, when pressed, increases the sound volume of all tibia ranks on all manuals. How would you make the equivalent of a voicing tab for jOrgan?

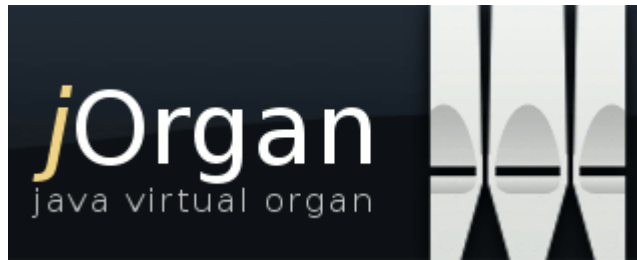
*Answer:* Make a voicing command with a switch filter, much as I did in the ACO 3P-49. The big Schlicker disposition has a “Neobaroque Voicing” switch that triggers several switch filters, bringing down the 16s and 8s and raising the 2s and mixtures.

Make a switch filter like this:

Engaged        set 176 set 7 set 120

Disengaged    set 176 set 7 set 100

Reference the switch filter to the desired ranks. “Disengaged” is the default position. “Engaged” would increase the volume. You can go up as high as set 127. [Paul 7/4/2010]



## 18. jOrgan Beginning Developers MIDI Language Corner

### Introduction

The jOrgan Beginner's earliest encounter with MIDI technology typically involves making MIDI connections between hardware elements, such as keyboards and computers. Later on, as a Beginner adapting other dispositions for his own use or taking the leap forward into developing his own dispositions, the Beginner comes up against the really abstruse innards of MIDI—the language itself.

In jOrgan one encounters MIDI language messages in at least three places:

- the messages section of a jOrgan element, see figure (to be added)
- the messages pane of the jOrgan GUI, see figure (to be added)
- the monitor pane of the jOrgan GUI, see figure (to be added).

It is natural for a Beginner to be intimidated by his first exposure to MIDI language messages, that is to coded instructions that are interpreted by MIDI devices and by jOrgan. That should not be the case, because the MIDI language itself is deliberately designed to be simple, and it is. The problem lies not in the language but in how the language is presented. This is not a fault of jOrgan but of the traditional manner in which what is essentially engineering specifications are presented to the general public. It is traditional for engineers and standards committees to use an *in medias res* approach to communicate their rules, which sort of leaves the Beginner confounded, and is certainly not helpful to people who feel they should understand something before they begin to use it.

In getting around the general problem of introducing an argot, such as MIDI specs, to people who need a working knowledge of it, not as an end in itself, but as a tool to solve other problems, I believe in an approach that begins right where the Beginner initially finds himself and proceeds from there to introduce whatever new definitions and facts as are needed at each step.

So, in the following sections, we will start off with what jOrgan sees in the Monitor pane under various conditions, see what those displays mean, then proceed from there to try modifying element messages and debugging messages of our own design and what we find in other designers' dispositions.

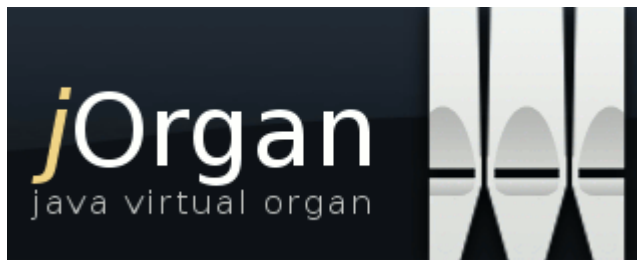
Along the way we will find out about the different kinds of MIDI commands and how they differ from MIDI data and, in passing, build up a modest acquaintance with binary, decimal and hexadecimal number systems and learn to distinguish between ordinal number systems and enumerative/calculational number systems, but only when we need it. Mainly we will try to stick with learning only what we need to know about MIDI's language to accomplish what we are trying to do with jOrgan.

## Seeing (And Modifying) MIDI—What MIDI Information Looks Like

The jOrgan Beginner has two different ways to actually look at MIDI information:

- as captured by the jOrgan Monitor Pane
- in user-written jOrgan messages
- as translate into text from .mid MIDI files.

Lynn Walls suggests the following for translating MIDI files: use the MIDI-File-to-Text utility (mf2t.exe) that translates the binary bit stream data of a midi file into human readable text, example at <http://dl.dropbox.com/u/6887947/orch.txt>. The companion program (t2mf.exe) translates MIDI text back into a standard MIDI file format. This pair of programs is really handy if one absolutely must edit a MIDI file—run mf2t to turn the MIDI file into a text file, edit the text file with a normal text editor (Notepad will do), then run t2mf to turn the edited text file back into a midi file. *Caution:* it is very easy to corrupt a midi file this way if you don't follow the rules when you edit the text file.



## 19. jOrgan Beginners Introduction to Extensions

### Introduction

This section is intended to explain the various extensions that have been provided as advanced features to jOrgan users and jOrgan disposition developers. Where possible actual operational dispositions will be used as examples of how these extensions are employed.

The official jOrgan site at <http://jorgan.sourceforge.net/introduction> lists the following extensions <http://jorgan.sourceforge.net/extension>:

- the customize feature—employed in Ch. 4 and Ch. 5, vol. 1
- the Creative Soundcards feature—described in Ch. 7, vol. 1
- ability to execute arbitrary commands
- the embed-able FluidSynth sampler—discussed in Ch. 7, vol. 1 and Ch. 12, vol. 2
- importing elements from other sources
- the jOrgan virtual keyboard—introduced in Ch.3 and Ch. 4, vol. 1
- connecting MIDI devices on other computers with LAN
- configuring Linuxsampler
- the Memory element and how to use it—discussed in Ch. 11, vol. 2
- Merging MIDI devices
- Recording jOrgan performances—described in Ch. 8, vol. 1
- SAMs-driven stop tabs
- using Soundfonts—described in Ch. 7, vol. 1
- MIDI tools and utilities.

(to be continued)

## **Creative Soundcards Feature**

(to be continued)

## **FluidSynth Sampler Feature**

(to be continued)

## **Import Elements Feature**

(to be continued)

## **Virtual Keyboard Feature**

(to be continued)

## **LAN Devices Feature**

(to be continued)

## **LinuxSampler Configure Feature**

(to be continued)

## **Memory Feature**

(to be continued)

## **Merge MIDI Devices Feature**

(to be continued)

## Record Performances Feature

jOrgan's Record Performances Feature allows the user to record his exact performance with a particular disposition, including stops and voices, expression and other console dynamics. Afterwards the .mid file which jOrgan produces can be used to replicate the performance exactly on that same disposition. How to add the recording element to a disposition and make recordings is described in Ch. 8, Vol. I of the Beginners Guide. The section you are reading now, on the other hand, deals with Record issues that a more advanced developer might encounter.

*Caution:* jOrgan's .mid file is intended for use only with jOrgan's own record-playback feature. If a jOrgan .mid file were played back on some other MIDI player—or if a person used JOrgan to play a .mid file from some source other than jOrgan—the results would probably be disappointing. This analysis from Lynn Walls explains why.

### WHY ONE SHOULD USE ONLY JORGAN .MID FILES FOR PLAYBACK WITH JORGAN

A jOrgan MIDI file follows the basic format of a standard midi file, and the Note-On and Note-Off messages for each Keyboard defined in jOrgan are recorded as true MIDI Note-On and Note-Off messages on separate tracks—one track for each Keyboard. But the Note-On/Off messages are all recorded as channel 0. Also, all the Stop/Piston controls are handled through SMF Meta Text messages (which are really not midi messages at all).

A midi file recorded/created by anything other than jOrgan itself MIGHT make some sound if loaded into the jOrgan player -- but it would not be what you want. You are better off playing such a midi file through a midi sequencer and directing the midi output of this sequencer to the Keyboard elements of an appropriately configured jOrgan disposition, connecting the sequencer MIDI device output to a jOrgan midi device input using one of the Virtual MIDI Patch Cable tools like MIDI-Yoke or Maple.

(As far as playing back a jOrgan MIDI file through another sequencer is concerned) every sequencer that I know of will COMPLETELY IGNORE the Meta Text messages produced by the jOrgan recorder. The jOrgan recorder will always RECORD the stops used. If a jOrgan-recorded MIDI file is played back via a sequencer into jOrgan the stop changes are ignored.

The only way a midi file played by a sequencer could ever change jOrgan stops would be if the Stop ELEMENTS in the jOrgan disposition were all configured with “Activate” and “Deactivate” messages corresponding to the actual midi messages within the midi file intended to make those stop changes—assuming, of course, that such stop-changing midi messages even existed in the midi file. These MUST be actual MIDI messages—not Meta Text data. The Meta Text data recorded by the jOrgan recorder will be totally ignored by a sequencer, thus no MIDI messages causing jOrgan stop changes will be created from the jOrgan Meta Text or sent by the sequencer to jOrgan via the Virtual MIDI Cable method.

Stop changes made by flipping stops on and off on a jOrgan console display will be recorded in the MIDI file that the jOrgan recorder produces. If they are turned on and off by actual MIDI messages in the midi file that have been matched to the “Activate” and “Deactivate” message specifications of each jOrgan Stop ELEMENT, then they will also be recorded as Meta Text data in the midi file produced by the jOrgan Recorder. The jOrgan recorder automatically embeds stop/piston changes in its output MIDI files.

One could edit a jOrgan recorded MIDI file, using an external MIDI file editor, but he would need to know the relevant XML element ID tags in the disposition file.

### **WHY ONE SHOULD USE ONLY jORGAN .MID FILES FOR PLAYBACK WITH jORGAN (CONTINUED)**

Meta Text representing stop changes and swell/expression changes are coded into the jOrgan midi file in the FIRST track. the ONLY actual midi messages appearing in a jOrgan MIDI file are the Note-On and Note-Off MIDI messages. Everything else—every other control: Stops, Combo pistons, couplers, swell expression—is handled with Meta Text data and is totally un-processable by any known midi sequencer. You will never find a ProgChange (x'C0') or a ControlChange (x'B0') or any other midi command message other than NoteOn (x'90') and NoteOff (x'80') in a jOrgan-produced MIDI file.

## SAMS-Driven Stop Tabs Feature

(to be continued)

## Soundfonts Usage Feature

(to be continued)

### Execute Arbitrary System Commands Feature (Executor)

The Executor feature is a jOrgan element into which you place a Windows command-prompt type sentence, the sort of sentence you would type in a MS-DOS window. This command is executed when you activate the element, for example when you press a piston associated with the command. Whether or not jOrgan executes a Windows system command is set in the Configuration menu.

*Example*—For his Clicquot disposition Jacques Levy includes this sentence in the Command property of an Executor element:

`C:/Program Files/Adobe/Reader 9.0/Reader/AcroRd32.exe IMSLP03814-Couperinorgue.pdf`

On the Clicquot Organ console the effect of pressing a piston assigned to that Executor element is to open a new window containing the PDF sheet music document IMSLP03814-Couperinorgue.pdf.

***Troubleshooting Note:*** if an Executor element is not working, check the View>Configuration menu to verify that “Allow” has been set.

## **MIDI Tools and Utilities Feature**

(to be continued)

# Index

## A

AGO 16.1  
American Guild of Organists 16.1  
Architecture of a jOrgan Disposition 17.1

## B

Burd, Bill ii

## C

Crescendo Pedals Programming 17.3

## D

Dietzer, Dan ii  
Dispositions  
  troubleshooting 17.1

## E

### Extensions

  arbitrary commands 19.1  
  Creative Soundcards i, 19.1, 19.2  
  customize feature 19.1  
  FluidSynth sampler 19.1  
  importing elements 19.1  
  Linuxsampler 19.1  
  Memory element 19.1  
  Merging MIDI devices 19.1  
  MIDI devices on other computers with LAN 19.1  
  MIDI tools and utilities 19.1  
  Recording jOrgan performances 19.1  
  SAMs-driven stop tabs 19.1  
  Soundfonts i, 19.1, 19.7  
  virtual keyboard 19.1

## F

## G

GTune 16.1

## H

Hardy, Joseph ii  
Hauptwerk-Like Couplers 17.3

## I

## J

## **K**

## **L**

LoopBack) 16.2  
LoopBe 16.2  
Lynn Walls ii

## **M**

Maple 16.2  
Meakin, Tony ii  
Meier, Sven ii  
MIDI Message Information  
    What MIDI Information Looks Like i, 18.2  
Midi Over Lan CP 16.2

## **N**

## **O**

Order of Divisions 16.1

## **P**

## **Q**

## **R**

Recorder Element—Using 17.3  
Reimer, John ii  
Rodgers 340 17.6

## **S**

Soundfont Importing 17.4  
Sound Source Planning 16.1

## **T**

Transpose Coupler Implementation 17.4  
troubleshooting dispositions 17.1

## **U**

Unison Off Implementation 17.5

## **V**

Virtual MIDI Cables 16.2  
Virtual MIDI Channels 16.2  
Vista 2  
Voicing Command Implementation 17.6

## **W**

Walls, Lynn ii  
Whatson, Rick ii  
Windows XP 2

## **X**

## **Y**

## **Z**





