

La norme Midi et JavaSound

V 1.0 - 14.2.2006 (update Fev. 07)

Jacques Ferber

**LIRMM - Université Montpellier II
161 rue Ada
34292 Montpellier Cedex 5**

**Email: ferber@lirmm.fr
Home page: www.lirmm.fr/~ferber**



La norme M.I.D.I.

Le **M**usical **I**nstrument **D**igital **I**nterface est conçu initialement comme un protocole de communication entre équipements musicaux.

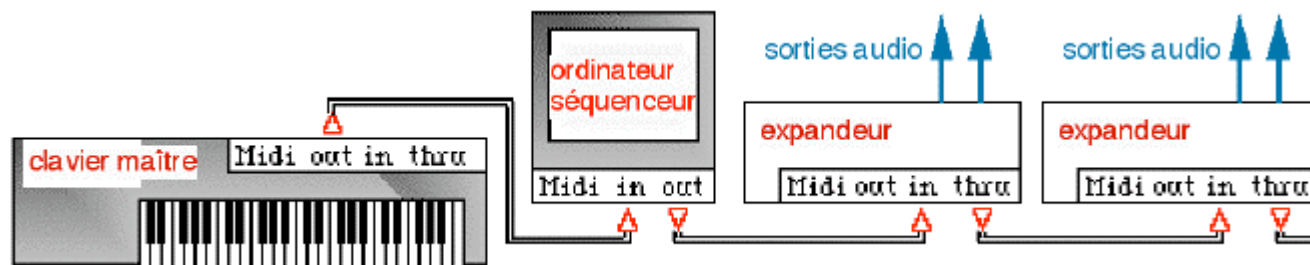
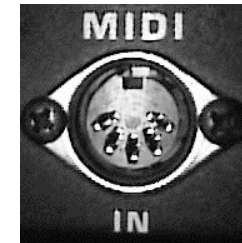
Origine: 1983.

L'information Midi qui circule entre 2 appareils est une suite de messages codées en quelques octets

Éléments de départ

Les connexions M.I.D.I.

- ◆ Des câbles Midi relient les différentes prises des appareils
- ◆ Les prises Midi sont généralement au nombre de trois: Midi In, Midi Out, Midi Thru
- ◆ Le standard Midi utilise un mode de transmission série asynchrone
- ◆ L'information circule à 31,25kbauds.



Configuration standard d'un équipement midi



Principe

◆ Des événements ont lieu sur le clavier

- Ces événements sont transmis aux expandeurs et/ou enregistrés (à une certaine vitesse/tempo) dans le séquenceur
- Le séquenceur peut rejouer ces événements (à une vitesse quelconque) comme s'il s'agissait du musicien qui jouait en direct.

◆ Les principaux événements midi

- Note on: une touche est enfoncée avec une certaine vitesse
- Note off: une touche est relâchée
- Aftertouch: pression sur une touche
- Program change: sélection d'un son sur un synthé (128 sons maxi)
- Control change: modification de paramètres sonores d'un synthé: volume, pan, sustain, brillance, résonance, etc..
- Midi clock: horloge de synchronisation entre 2 séquenceurs midi
- System Exclusifs (SysEx): code propre à chaque constructeur permettant des codage en profondeur des synthés et notamment sert au chargement de banques de son



Canaux midi

◆ La plupart des événements sont envoyés sur des canaux midi (on parle de « messages canaux »)

- Idée: chaque expandeur dispose de son canal midi
- Il y a 16 canal midi
 - ☞ A la fois beaucoup et peu si l'on songe que la plupart des synthés sont multi-timbres:
 - Ex: XV5080: peut jouer 128 voix sur 32 parties.. *Il fait déjà plus que la norme Midi*
 - Rajoutez quelques autres synthés: la norme midi est totalement dépassée...

◆ Les messages systèmes n'ont pas de canaux

- Midi clock (et apparentés)
 - ☞ Start et stop + gestion des top midi..
- SysEx

Les messages canaux

◆ Codage:

- 2 ou 3 octets
- 1^{er} octet:
 - ☞ Bit 1: statut message (octet de statut)
 - ☞ Bits 2 à 4: type de message
 - ☞ Bits 5 à 8 : canal
- 2^{ème} et 3^{ème} octet
 - ☞ Bit 1: statut donnée
 - ☞ Autres: données (128 valeurs donc)

◆ Les 7 messages canaux:

- NoteOff: 1000 nnnn + hauteur note + vitesse (?)
- NoteOn: 1001 nnnn + hauteur note + vitesse
- Polyphonic after touch: 1010 nnnn + Hauteur note + valeur pression
- After touch: 1101 nnnn + valeur:
 - ☞ variation de pression sur tout le clavier
- Pitch bend: 1110 nnnn + Code + Vitesse
- Program change: 1100 nnnn + Affectation d'un instrument à un canal
(=> 128 instruments seulement)
- Control Change: 1011 nnnn + Numéro + Valeur

Contrôleurs Midi: Control Change

CC	Contrôleur	CC	Contrôleur	CC	Contrôleur	CC	Contrôleur
0	Bank select (MSB)	32	Bank select (LSB)	64	Sustain pedal	96	
1	Modulation	33	Modulation	65	Porta pedal	97	
2	Breath	34		66	Sostenuto pedal	98	
3		35	Aftertouch	67	Soft pedal	99	
4	Foot control	36		68	Legato footswitch	100	
5	Portamento	37		69	Hold pedal 2	101	
6		38		70		102	
7	Volume	39		71	Resonance	103	
8	Balance	40		72	Release time	104	
9		41		73	Attack time	105	
10	Pan	42		74	LPF Cutoff	106	
11	Expression	43		75		120	All Sound off
						121	Reset all controls.
				91	Reverb		
				92	Tremolo depth		
				93	Chorus		
				94	Detune		

System exclusif

- ◆ **Propre à chaque fabricant**
- ◆ **Pas de longueur défini (nécessite un fin de message)**
- ◆ **Codage:**
 - 1111 0000 (début SysEx)
 - 0xxx xxxx : num ID constructeur. Nb octet variable, selon fabricant. (ex: Roland = 41H)
 - Réservé: 01111 1101
 - ... contenu
 - 1111 0111 (fin SysEx)



General Midi

◆ Codage des instruments

◆ GM1, 1991

- Table d'instruments de 128 instruments + 47 kits de percussions
- Canal 10 pour les percus.
- Gestion de contrôleurs midi de base
- Table d'instrument:

☞ Ex: 1. Piano , 5 Piano electr., 13 Marimba, 25. Guitare nylon,
34. Basse electr...58 Trombone, 128 coup de feu, etc..

◆ Existe maintenant la norme GM2, mais la plupart des fichiers midi sont encore en GM1

◆ Fichier midi: fichier contenant des événements midi qui peuvent être joués par des instruments

- (généralement GM1, mais pas toujours)



Java Sound

- ◆ **API de bas niveau permettant de contrôler l'input et output d'informations sonores**
 - Lire, capturer, mixer du son
 - ☞ Packages `javax.sound.sampled`
 - Gestion d'informations MIDI
 - ☞ Package `javax.sound.midi`
- ◆ **On s'intéressera uniquement aux aspects Midi de l'implémentation**

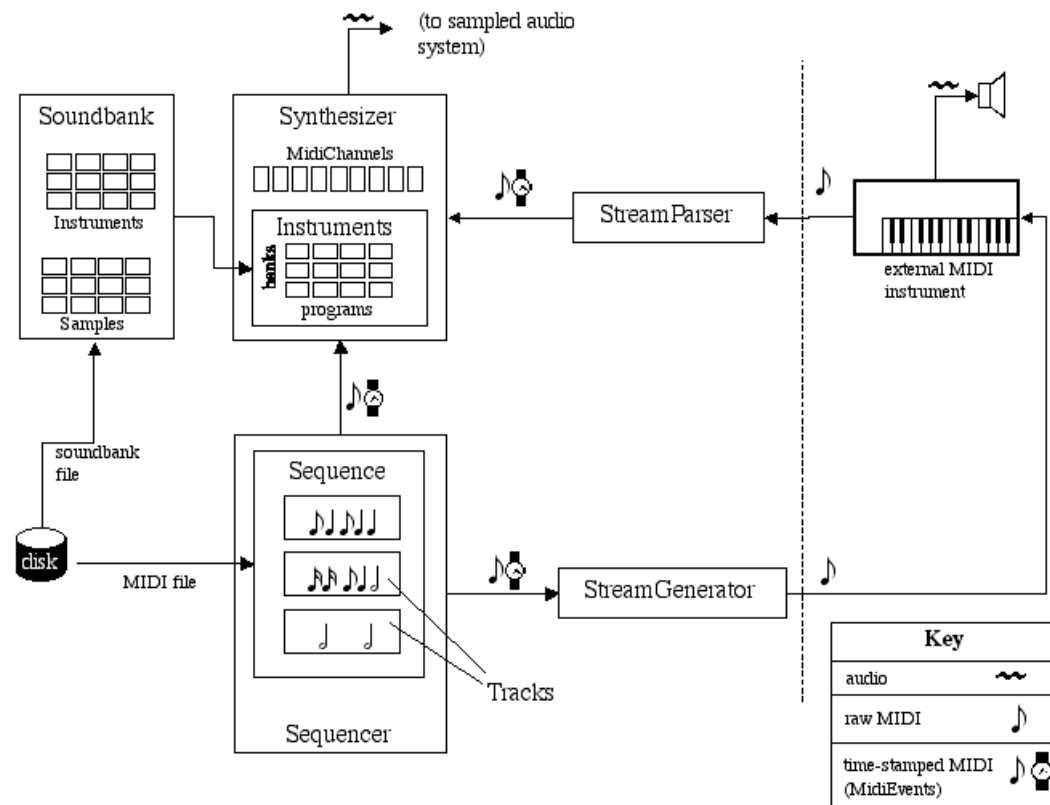


Java Sound Midi

- ◆ **Fournit un ensemble de mécanismes permettant de créer des applications MIDI en Java**
 - Pendant longtemps, il n'était pas possible de connecter à des instruments/claviers extérieurs
 - ☞ Limitation à des exemples jouets
 - Quelques problèmes de gestion du temps à cause d'un bug dans le JDK (et pas le JRE..)
- ◆ **Depuis la version 1.5, ces limitations ont été levées**
 - Entre-temps Tritonus a produit une implémentation efficace de JavaSound (cf. www.jsresources.org)

JavaSound API Midi

- ◆ Idée: connecter des appareils virtuels comme s'il s'agissait d'une configuration matérielle





Concepts

◆ **MidiMessage, MidiEvent**

- Les messages Midi de base: données midi + longueur des données
- MidiEvent: contient des messages midi avec une date en « ticks » en plus

◆ **MidiDevice (interface)**

- l'interface de base pour tous les appareils Midi
- Les modules de base sont: Sequencer, Synthesizer, et éventuellement d'autres...
- Pour le manipuler il faut lui adjoindre des « receiver » et des « transmitter » qui font office ports d'entrée-sortie.

◆ **Sequencer**

- Un sequencer est un appareil (device) qui sait enregistrer et lire une séquence (sequence)
- Une séquence est composée d'un ensemble de pistes (track)
- Un track comprend un ensemble de MidiEvent

◆ **Synthesizer**

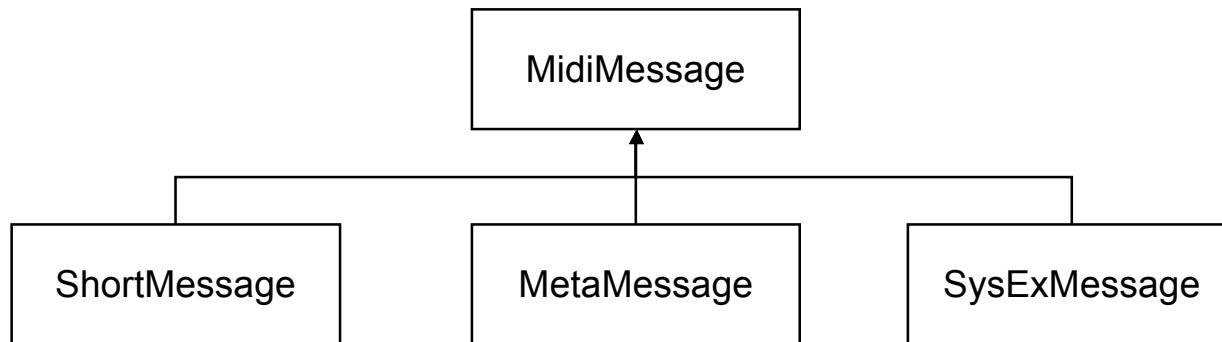
- Produit du son
- On peut définir des « soundbank » et jouer des sons particuliers..

- ◆ *Quelques bugs: l'implémentation de 1.3/1.4 considère que les sequencers sont aussi des synthétiseurs. A disparu (semble-t-il avec la version 1.5)*

MidiMessage

◆ MidiMessage

- Classe abstraite: Représentation d'un message de base Midi
- ShortMessage: note on, off, cc, program change, etc..
- MetaMessage: pour être utilisé par un séquenceur, ou autre..



MidiSystem

◆ La plaque tournante pour obtenir les ressources du système

◆ Toutes les méthodes sont 'static' public static

● MidiDevice.Info[] getMidiDeviceInfo()

☞ retourne un tableau d'infos sur les appareils (device) en cours:

```
0 OUT Mappeur MIDI Microsoft, Unknown vendor, 5.0, Windows MIDI_MAPPER
1 OUT Synthétiseur logiciel Microsoft, Unknown vendor, 5.10, Internal software synthesizer
2 IN OUT Real Time Sequencer, Sun Microsystems, Version 1.0, Software sequencer
3 OUT Java Sound Synthesizer, Sun Microsystems, Version 1.0, Software wavetable synthesizer and receiver
```

● MidiDevice getMidiDevice(MidiDevice.Info info)

☞ Récupère l'appareil à partir de son nom.. ATTENTION: c'est le seul moyen pour récupérer des devices qui ne sont pas par défaut

● Synthesizer getSynthesizer()

☞ Retourne le synthétiseur par défaut

● Sequencer getSequencer()

☞ Retourne le séquenceur par défaut

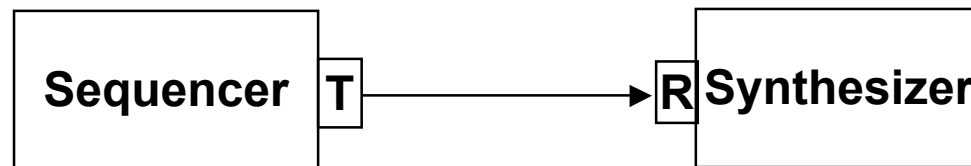
● Sequence getSequence(File file)

☞ Retourne une séquence à partir du fichier MIDI

Connexion d'appareils

◆ Connecte les devices par leur connecteurs

- Deux types de connecteurs: Receiver (input) et Transmitter (out)
- On peut associer un 'Transmitter' à un 'Receiver'



```
Receiver synthReceiver = monSynthe.getReceiver();  
Transmitter seqTransmitter = monSeq.getTransmitter();  
seqTransmitter.setReceiver(synthReceiver);
```



Sequencer

◆ Lire, arrêter une séquence

- start(), stop()
- Placer une séquence: setSequence(Sequence sequence)
- isRunning: indique si le séquenceur est en route
- setTrackMute(int track, boolean mute), getTrackMute(int track)
- setTrackSolo(int track, boolean solo)

◆ Enregistrement

- recordEnable(Track track, int channel): prépare la piste en question qui est prêt à recevoir des événements midi
- recordDisable(Track track)
- startRecording(), isRecording()

◆ Tempo

- setTempo(float bpm), getTempo(float bpm)

◆ Plus:

- Opérations de bouclage (loop) de parties de séquences
- Synchronisations (master, slave) avec la midi clock (interne ou externe, dépendant du SyncMode du sequencer..)
- Positionnement en tick dans la séquence (dépend du tempo et de la résolution du tick)



Synthesizer

◆ **Produit du son**

◆ **Gestion des canaux:**

- MidiChannel[] getChannels()

◆ **On peut charger des Soundbank (banque de sons)**

- Soundbank getDefaultSoundbank()
- Instrument[] getAvailableInstruments()
- Instrument[] getLoadedInstruments()