Sampled Waveforms And Musical Instruments

Software for creating, editing and managing digital audio instruments.

Author:
Josh Green

CAUTION!
ACHTUNG!

The presentation you are about to see may contain unstable code, incomplete features and unsettled application programming interfaces. While we are confident that the highest quality care and determination has been applied to this project, we cannot be held responsible for any loss of sanity it may cause.

Presentation Overview

- About project Swami
- Instrument format comparison
- Swami architecture
- Python bindings
- Making instrument design easy
- CRAM instrument compression format
- PatchesDB instrument database
- Interfacing with other projects
- Software demo
- Questions and comments

Project: Swami

What it is...

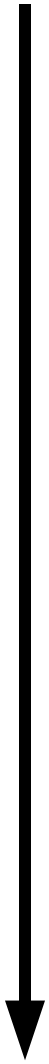
- Instrument editor/manager/librarian
- Cross platform (Linux, OSX, Win32)
- Re-usable software libraries
- Instrument compression technology
- Python script-able

What it isn't...

- Audio editing application
- Software synthesizer
- MIDI sequencer

Hint: Useful external software to interface with

Project time line



- 1999: Smurf SoundFont Editor publicly released.
- 2002: Renamed to Swami after re-write to abstract instrument logic from GUI and move from hardware AWE driver to software synthesizer FluidSynth.
- 2003: Forked Swami development branch to use GObject with instrument logic in separate library libInstPatch and port to GTK2.
- 2003: CRAM (Compress hybRid Audio Media) project started to create a standard for open instrument compression using FLAC.
- Nov, 2003: Last version of 0.9.x branch released (0.9.2), no longer being worked on.
- Near future, year 2006: Development branch is released as Swami 1.0.0.

Instrument Formats

- Focus on sample based “wavetable” instruments
- SoundFont
- DLS
- GigaSampler
- Others (Akai, create new format?)

Format Features

SoundFont

- 2 Envelopes, 2 LFOs, Low Pass Filter, Reverb, Chorus
- 16 bit mono/stereo and 24 bit (ver. 2.04)
- Modulators for real-time effect control
- 3 tier tree: Preset, Instrument, Sample
- 16 bit synthesis parameter resolution

DLS

- 2 Envelopes, 2 LFOs, Low Pass Filter, Reverb, Chorus
- 8/16 bit mono/stereo/surround - theoretically any WAV format
- Modulators - only a subset of effects is required for compliance
- 2 tier tree: Instrument, Sample
- 32 bit synthesis parameter resolution

GigaSampler

- 3 Envelopes, 3 LFOs, 4 filter types, many other misc. features
- 16/24 bit mono/stereo – others?
- Effect modulation
- 2 tier tree: Instrument, Sample with dimensions
- 8 or 16 bit synthesis parameter resolution

Format Pros & Cons

SoundFont

- Pros
  - Very popular
  - Open standard
  - Backwards compatible 24 bit samples
  - Modulate most effects
- Cons
  - File format not flexible
  - 24 bit is a new extension
  - 2 GB RIFF file limit

DLS

- Pros
  - Open standard
  - Flexible file format
  - Custom extensions
  - Large 32 bit parameter ranges
- Cons
  - Ver. 2 docs must be purchased
  - > 16 bit audio not part of spec
  - Modulating many effects not required for compliance
  - 2 GB RIFF file limit

GigaSampler

- Pros
  - Popular
  - 24 bit sample supp.
  - Many nice synthesis innovations (dimensions, cross fading, etc)
  - Can be multiple files (> 2GB)
- Cons
  - Proprietary
  - File format not flexible ??
  - Small parameter ranges

Order of increasing dependence

Swami Architecture

- C code using GObject and GTK2 (GUI only)
- libInstPatch
  - File <--> Objects (loading/saving)
  - Sample format conversion
  - Instrument object conversions
  - Object paste system
  - CRAM compression
- libSwami
  - GValue control networks
  - Wavetable drivers
  - Plugins (FluidSynth, FFTune)
- libSwamiGUI
  - Object oriented (drag-n-drop interfaces, subdivide views)
  - Model/view/controller

Python bindings

- Generated using PyGTK scripts
- Really OO - no refcounting or macro casts
- Built-in Python editor in SwamiGUI

Making instrument design easy

- Obtain samples
  - Slice up recorded material
  - Existing samples
  - Synthesized audio
- Define loops for continuous samples
  - Auto loop finder
  - Cross fader
- Tune
  - FFTune – Plugin using fftw library
  - Play along side reference tones
- Assign samples to optimal instrument key splits

CRAM instrument compression

- Compress hybRid Audio Media (CRAM)
- Based on EBML (like XML but binary)
- Format is expandable
- Uses FLAC for audio and zlib for binary
- Multi-file archives with timestamp info
- Format specific encoders but decoder is generic
- Future: Lossy Vorbis audio compression

PatchesDB instrument database

- WWW instrument database
- User accounts for contributing content
- CRAM used for compression
- Written in Python using MySQL
- Supports libInstPatch instrument formats
- Future: Pre-listen via streaming audio, browse/synch/download with local Swami

Interfacing with other Software

- Soft synths (LinuxSampler)
- Swami GStreamer plugin
- Audio editors
- DSSI synth plugin
- Sequencers (DSSI and ALSA seq.)

Help wanted

- Coders
- Web designers
- Graphic and icon artists