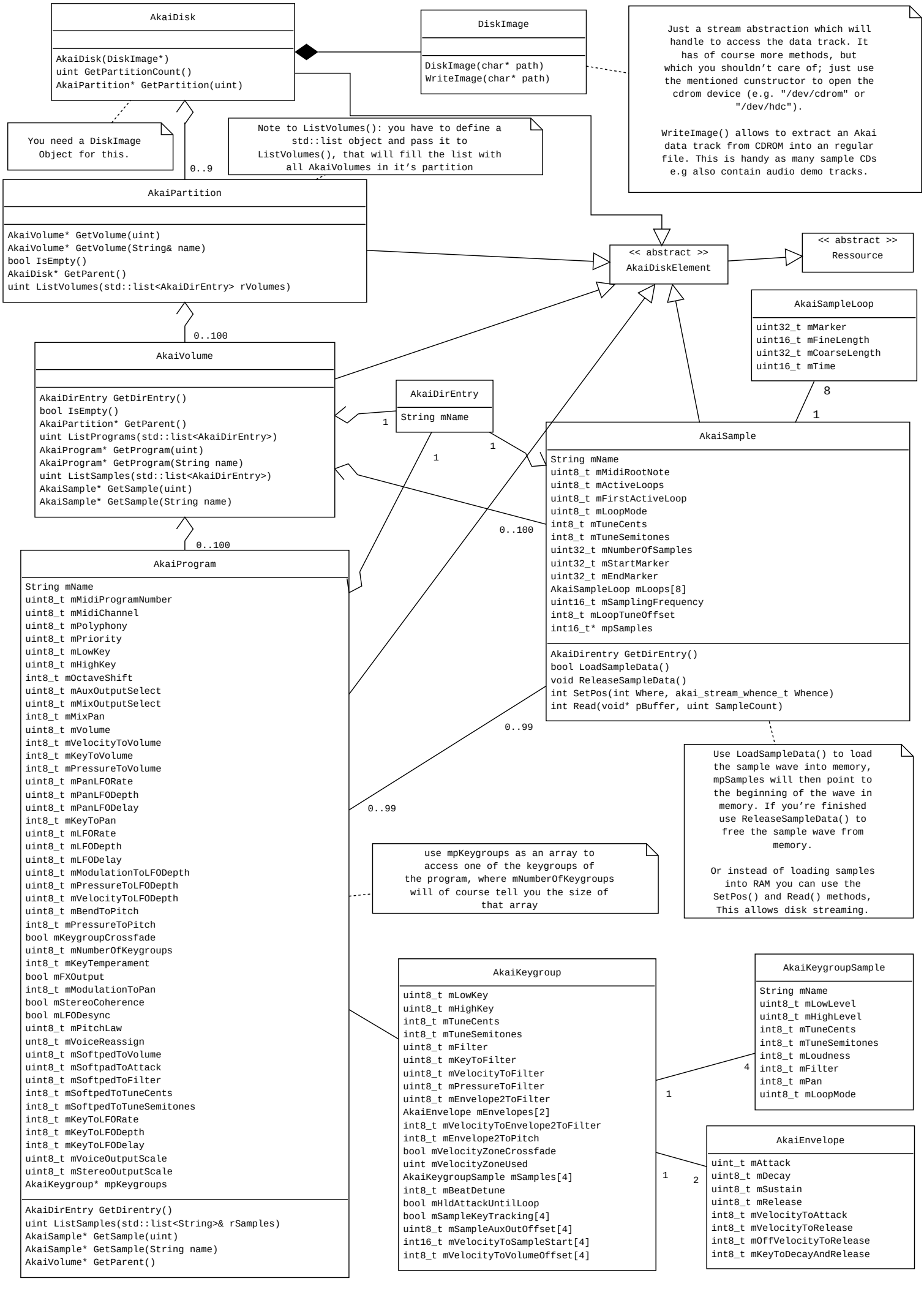
AkaiDisk
AkaiDisk(DiskImage*)
uint GetPartitionCount()
AkaiPartition* GetPartition(uint)
DiskImage
DiskImage(char* path)
WriteImage(char* path)
Just a stream abstraction which will handle to access the data track. It has of course more methods, but which you shouldn't care of; just use the mentioned cunstructor to open the cdrom device (e.g. "/dev/cdrom" or "/dev/hdc").
WriteImage() allows to extract an Akai data track from CDROM into an regular file. This is handy as many sample CDs e.g also contain audio demo tracks.
You need a DiskImage Object for this.
Note to ListVolumes(): you have to define a std::list object and pass it to ListVolumes(), that will fill the list with all AkaiVolumes in it's partition
0..9
AkaiPartition
AkaiVolume* GetVolume(uint)
AkaiVolume* GetVolume(String& name)
bool IsEmpty()
AkaiDisk* GetParent()
uint ListVolumes(std::list<AkaiDirEntry> rVolumes)
<< abstract >>
AkaiDiskElement
<< abstract >>
Ressource
0..100
AkaiVolume
AkaiDirEntry GetDirEntry()
bool IsEmpty()
AkaiPartition* GetParent()
uint ListPrograms(std::list<AkaiDirEntry>)
AkaiProgram* GetProgram(uint)
AkaiProgram* GetProgram(String name)
uint ListSamples(std::list<AkaiDirEntry>)
AkaiSample* GetSample(uint)
AkaiSample* GetSample(String name)
AkaiSampleLoop
uint32_t mMarker
uint16_t mFineLength
uint32_t mCoarseLength
uint16_t mTime
8
1
AkaiDirEntry
String mName
1
1
1
AkaiSample
String mName
uint8_t mMidiRootNote
uint8_t mActiveLoops
uint8_t mFirstActiveLoop
uint8_t mLoopMode
int8_t mTuneCents
int8_t mTuneSemitones
uint32_t mNumberOfSamples
uint32_t mStartMarker
uint32_t mEndMarker
AkaiSampleLoop mLoops[8]
uint16_t mSamplingFrequency
int8_t mLoopTuneOffset
int16_t* mpSamples
AkaiDirentry GetDirEntry()
bool LoadSampleData()
void ReleaseSampleData()
int SetPos(int Where, akai_stream_whence_t Whence)
int Read(void* pBuffer, uint SampleCount)
0..100
0..100
AkaiProgram
String mName
uint8_t mMidiProgramNumber
uint8_t mMidiChannel
uint8_t mPolyphony
uint8_t mPriority
uint8_t mLowKey
uint8_t mHighKey
int8_t mOctaveShift
uint8_t mAuxOutputSelect
uint8_t mMixOutputSelect
int8_t mMixPan
uint8_t mVolume
int8_t mVelocityToVolume
int8_t mKeyToVolume
int8_t mPressureToVolume
uint8_t mPanLFORate
uint8_t mPanLFODepth
uint8_t mPanLFODelay
int8_t mKeyToPan
uint8_t mLFORate
uint8_t mLFODepth
uint8_t mLFODelay
uint8_t mModulationToLFODepth
uint8_t mPressureToLFODepth
uint8_t mVelocityToLFODepth
uint8_t mBendToPitch
int8_t mPressureToPitch
bool mKeygroupCrossfade
uint8_t mNumberOfKeygroups
int8_t mKeyTemperament
bool mFXOutput
int8_t mModulationToPan
bool mStereoCoherence
bool mLFODesync
uint8_t mPitchLaw
unt8_t mVoiceReassign
uint8_t mSoftpedToVolume
uint8_t mSoftpadToAttack
uint8_t mSoftpedToFilter
int8_t mSoftpedToTuneCents
int8_t mSoftpedToTuneSemitones
int8_t mKeyToLFORate
int8_t mKeyToLFODepth
int8_t mKeyToLFODelay
uint8_t mVoiceOutputScale
uint8_t mStereoOutputScale
AkaiKeygroup* mpKeygroups
AkaiDirEntry GetDirentry()
uint ListSamples(std::list<String>& rSamples)
AkaiSample* GetSample(uint)
AkaiSample* GetSample(String name)
AkaiVolume* GetParent()
0..99
0..99
Use LoadSampleData() to load the sample wave into memory, mpSamples will then point to the beginning of the wave in memory. If you're finished use ReleaseSampleData() to free the sample wave from memory.
Or instead of loading samples into RAM you can use the SetPos() and Read() methods, This allows disk streaming.
use mpKeygroups as an array to access one of the keygroups of the program, where mNumberOfKeygroups will of course tell you the size of that array
AkaiKeygroup
uint8_t mLowKey
uint8_t mHighKey
int8_t mTuneCents
int8_t mTuneSemitones
uint8_t mFilter
uint8_t mKeyToFilter
uint8_t mVelocityToFilter
uint8_t mPressureToFilter
uint8_t mEnvelope2ToFilter
AkaiEnvelope mEnvelopes[2]
int8_t mVelocityToEnvelope2ToFilter
int8_t mEnvelope2ToPitch
bool mVelocityZoneCrossfade
uint mVelocityZoneUsed
AkaiKeygroupSample mSamples[4]
int8_t mBeatDetune
bool mHldAttackUntilLoop
bool mSampleKeyTracking[4]
uint8_t mSampleAuxOutOffset[4]
int16_t mVelocityToSampleStart[4]
int8_t mVelocityToVolumeOffset[4]
AkaiKeygroupSample
String mName
uint8_t mLowLevel
uint8_t mHighLevel
int8_t mTuneCents
int8_t mTuneSemitones
int8_t mLoudness
int8_t mFilter
int8_t mPan
uint8_t mLoopMode
4
1
1
2
AkaiEnvelope
uint_t mAttack
uint8_t mDecay
uint8_t mSustain
uint8_t mRelease
int8_t mVelocityToAttack
int8_t mVelocityToRelease
int8_t mOffVelocityToRelease
int8_t mKeyToDecayAndRelease