

Bse API Reference

0.15.0

The Beast Project <beast.testbit.org>

November 2019

Contents

1	Namespace Documentation	6
1.1	Bse Namespace Reference	6
1.2	Bse::AnsiColors Namespace Reference	47
1.3	Bse::Lib Namespace Reference	49
1.4	Bse::Path Namespace Reference	49
1.5	Bse::Procedure Namespace Reference	54
1.6	Bse::Re Namespace Reference	54
1.7	Bse::Test Namespace Reference	54
1.8	Bse::Xms Namespace Reference	56
1.9	Sfi Module Reference	56
2	Class Documentation	58
2.1	Bse::AlignedArray< T, ALIGNMENT > Class Template Reference	58
2.2	Bse::AlignedPOD< SIZE > Struct Template Reference	58
2.3	Bse::AsyncBlockingQueue< Value > Class Template Reference	58
2.4	Bse::AutoSeeder Class Reference	59
2.5	Bse::AuxData Struct Reference	60
2.6	Bse::AuxDataSeq Struct Reference	60
2.7	Bse::Blob Class Reference	60
2.8	Bse::Bus Interface Reference	62
2.9	Bse::Busiface Class Reference	64
2.10	Bse::BusImpl Class Reference	66
2.11	Bse::Category Struct Reference	66
2.12	Bse::CategorySeq Struct Reference	67
2.13	Bse::Container Interface Reference	67
2.14	Bse::ContainerIface Class Reference	68
2.15	Bse::ContainerImpl Class Reference	70
2.16	Bse::ContextMerger Interface Reference	70
2.17	Bse::ContextMergerIface Class Reference	71
2.18	Bse::ContextMergerImpl Class Reference	73
2.19	ConvertAny Struct Reference	74
2.20	Bse::CSynth Interface Reference	74
2.21	Bse::CSynthIface Class Reference	75
2.22	Bse::CSynthImpl Class Reference	76
2.23	Bse::Xms::DataConverter< T, typename > Struct Template Reference	76
2.24	Bse::DataKey< Type > Class Template Reference	77
2.25	Bse::DataList Class Reference	77
2.26	Bse::DataListContainer Class Reference	78
2.27	Bse::Device Interface Reference	80
2.28	Bse::DeviceCrawlerIface Class Reference	82
2.29	Bse::DeviceCrawlerImpl Class Reference	83
2.30	Bse::DeviceIface Class Reference	83
2.31	Bse::DeviceImpl Class Reference	85
2.32	Bse::DeviceInfo Struct Reference	86
2.33	Bse::DriverEntry Struct Reference	86
2.34	Bse::DriverEntrySeq Struct Reference	86
2.35	Bse::EditableSample Interface Reference	86
2.36	Bse::EditableSampleIface Class Reference	89

2.37	Bse::EditableSampleImpl Class Reference	90
2.38	Bse::Flac1Handle Class Reference	91
2.39	Bse::FloatSeq Struct Reference	92
2.40	Bse::FriendAllocator< T > Class Template Reference	92
2.41	Bse::Icon Struct Reference	93
2.42	Bse::Item Interface Reference	94
2.43	Bse::ItemIface Class Reference	98
2.44	Bse::ItemImpl Class Reference	100
2.45	Bse::ItemSeq Struct Reference	102
2.46	Bse::KeccakCryptoRng Class Reference	102
2.47	Bse::Lib::KeccakF1600 Class Reference	104
2.48	Bse::KeccakFastRng Class Reference	105
2.49	Bse::KeccakGoodRng Class Reference	106
2.50	Bse::KeccakRng Class Reference	108
2.51	Bse::LegacyObject Interface Reference	112
2.52	Bse::LegacyObjectIface Class Reference	114
2.53	Bse::LegacyObjectImpl Class Reference	116
2.54	Bse::MidiNotifier Interface Reference	116
2.55	Bse::MidiNotifierIface Class Reference	118
2.56	Bse::MidiNotifierImpl Class Reference	119
2.57	Bse::MidiSynth Interface Reference	119
2.58	Bse::MidiSynthIface Class Reference	121
2.59	Bse::MidiSynthImpl Class Reference	122
2.60	Bse::Module Class Reference	122
2.61	Bse::ModuleIface Class Reference	124
2.62	Bse::ModuleImpl Class Reference	126
2.63	Bse::ModuleTypeInfo Struct Reference	127
2.64	Bse::NoteDescription Struct Reference	127
2.65	Bse::Object Interface Reference	127
2.66	Bse::ObjectIface Class Reference	129
2.67	Bse::ObjectImpl Class Reference	130
2.68	Bse::Part Interface Reference	130
2.69	Bse::PartControl Struct Reference	137
2.70	Bse::PartControlSeq Struct Reference	137
2.71	Bse::PartIface Class Reference	138
2.72	Bse::PartImpl Class Reference	139
2.73	Bse::PartLink Struct Reference	140
2.74	Bse::PartLinkSeq Struct Reference	140
2.75	Bse::PartNote Struct Reference	140
2.76	Bse::PartNoteSeq Struct Reference	140
2.77	Bse::PartSeq Struct Reference	140
2.78	Bse::Pcg32Rng Class Reference	141
2.79	Bse::PcmWriter Interface Reference	142
2.80	Bse::PcmWriterIface Class Reference	144
2.81	Bse::PcmWriterImpl Class Reference	145
2.82	Bse::PixelSeq Struct Reference	145
2.83	Bse::ProbeFeatures Struct Reference	146
2.84	Bse::Project Interface Reference	146
2.85	Bse::ProjectIface Class Reference	153
2.86	Bse::ProjectImpl Class Reference	155
2.87	Bse::PropertyCandidates Struct Reference	156
2.88	Bse::Xms::SerializationNode::QueuedArgs Struct Reference	156
2.89	Bse::Xms::Reflink Class Reference	156
2.90	Bse::Resampler2 Class Reference	156
2.91	Bse::SampleFileInfo Struct Reference	158
2.92	Bse::Lib::ScopedLocale Class Reference	159
2.93	Bse::Lib::ScopedPosixLocale Class Reference	159
2.94	Bse::Sequencer Class Reference	160
2.95	Bse::Xms::SerializableInterface Class Reference	161

2.96	Bse::Xms::SerializationField Class Reference	161
2.97	Bse::Xms::SerializationNode Class Reference	163
2.98	Bse::Server Interface Reference	167
2.99	Bse::ServerIface Class Reference	175
2.100	Bse::ServerImpl Class Reference	177
2.101	SfiRecFields Struct Reference	178
2.102	Bse::SHA3_224 Struct Reference	178
2.103	Bse::SHA3_256 Struct Reference	179
2.104	Bse::SHA3_384 Struct Reference	180
2.105	Bse::SHA3_512 Struct Reference	181
2.106	Bse::SHAKE128 Struct Reference	182
2.107	Bse::SHAKE256 Struct Reference	183
2.108	Bse::SharedMemory Struct Reference	184
2.109	Bse::ShmFragment Struct Reference	185
2.110	Bse::ShmFragmentSeq Struct Reference	185
2.111	Bse::SignalMonitor Interface Reference	185
2.112	Bse::SignalMonitorIface Class Reference	188
2.113	Bse::SignalMonitorImpl Class Reference	189
2.114	Bse::SNet Interface Reference	189
2.115	Bse::SNetIface Class Reference	191
2.116	Bse::SNetImpl Class Reference	193
2.117	Bse::Song Interface Reference	194
2.118	Bse::SongIface Class Reference	197
2.119	Bse::SongImpl Class Reference	199
2.120	Bse::SongTiming Struct Reference	200
2.121	Bse::SoundFont Interface Reference	200
2.122	Bse::SoundFontIface Class Reference	201
2.123	Bse::SoundFontImpl Class Reference	202
2.124	Bse::SoundFontRepoIface Class Reference	202
2.125	Bse::SoundFontRepoImpl Class Reference	204
2.126	Bse::Source Interface Reference	204
2.127	Bse::SourceIface Class Reference	210
2.128	Bse::SourceImpl Class Reference	212
2.129	Bse::Spinlock Class Reference	212
2.130	Bse::Lib::StringFormatter Class Reference	213
2.131	Bse::Strings Class Reference	214
2.132	Bse::StringSeq Struct Reference	215
2.133	Bse::SubSynth Interface Reference	215
2.134	Bse::SubSynthIface Class Reference	217
2.135	Bse::SubSynthImpl Class Reference	218
2.136	Bse::Super Interface Reference	219
2.137	Bse::SuperIface Class Reference	219
2.138	Bse::SuperImpl Class Reference	221
2.139	Bse::SuperSeq Struct Reference	221
2.140	Bse::TaskRegistry Class Reference	222
2.141	Bse::TaskStatus Struct Reference	223
2.142	Bse::Test::Timer Class Reference	225
2.143	Bse::Track Interface Reference	226
2.144	Bse::TrackIface Class Reference	230
2.145	Bse::TrackImpl Class Reference	232
2.146	Bse::TrackPart Struct Reference	233
2.147	Bse::TrackPartSeq Struct Reference	233
2.148	Bse::TrackSeq Struct Reference	233
2.149	Bse::ItemImpl::UndoDescriptor< Obj > Class Template Reference	233
2.150	Bse::UserMessage Struct Reference	233
2.151	Bse::Wave Interface Reference	235
2.152	Bse::WaveIface Class Reference	236
2.153	Bse::WaveImpl Class Reference	238
2.154	Bse::WaveOsc Interface Reference	239

2.155	Bse::WaveOscIface Class Reference	240
2.156	Bse::WaveOscImpl Class Reference	242
2.157	Bse::WaveOscSeq Struct Reference	242
2.158	Bse::WaveRepo Interface Reference	243
2.159	Bse::WaveRepoIface Class Reference	244
2.160	Bse::WaveRepoImpl Class Reference	246
3	File Documentation	247
3.1	bse/bseapi.idl File Reference	247
	Bibliography	251
	Index	252

Chapter 1

Namespace Documentation

1.1 Bse Namespace Reference

The `Bse` namespace contains all functions of the synthesis engine.

Namespaces

- [AnsiColors](#)
The `AnsiColors` namespace contains utility functions for colored terminal output.
- [Lib](#)
Namespace for implementation internals.
- [Path](#)
The `Path` namespace provides functions for file path manipulation and testing.
- [Procedure](#)
The `Procedure` namespace contains procedure/IDL helpers.
- [Re](#)
Some `std::regex` wrappers to simplify usage and reduce compilation time.
- [Test](#)
The `Test` namespace offers utilities for unit tests.
- [Xms](#)
Namespace for XML based serialization.

Classes

- class [AlignedArray](#)
Class to maintain an array of aligned memory.
- struct [AlignedPOD](#)
Helper to provide memory for placement new `AlignedPOD<SIZE>` is aligned like `max_align_t` or like `malloc()`-ed memory and provides `SIZE` bytes.
- class [AsyncBlockingQueue](#)
Asynchronous queue to push/pop values across thread boundaries.
- class [AutoSeeder](#)
`AutoSeeder` provides non-deterministic seeding entropy.
- struct [AuxData](#)
`AuxData` - record to describe entity attributes with "key = value" strings.
- struct [AuxDataSeq](#)
`AuxDataSeq` - a variable length list of `AuxData` records.
- class [Blob](#)
Binary large object storage container.
- interface [Bus](#)
Interface for effect stacks and per-track audio signal routing to the master output.

- class [BusIface](#)
IDL interface class for `Bse::Bus`.
- class [BusImpl](#)
- struct [Category](#)
Categories describe useful type entities.
- struct [CategorySeq](#)
Sequence of `Category` records.
- interface [Container](#)
Base interface type for containers of `Item` derived types.
- class [ContainerIface](#)
IDL interface class for `Bse::Container`.
- class [ContainerImpl](#)
- interface [ContextMerger](#)
Source module for merging multiple synthesis contexts, used to implement polyphony.
- class [ContextMergerIface](#)
IDL interface class for `Bse::ContextMerger`.
- class [ContextMergerImpl](#)
- interface [CSynth](#)
Customizable synthesis (filter) network container.
- class [CSynthIface](#)
IDL interface class for `Bse::CSynth`.
- class [CSynthImpl](#)
- class [DataKey](#)
`DataKey` objects are used to identify and manage custom data members of `DataListContainer` objects.
- class [DataList](#)
Underlying storage implementation for a `DataListContainer`.
- class [DataListContainer](#)
`DataListContainer` - typesafe storage and retrieval of arbitrary members.
- interface [Device](#)
Interface for the encapsulation of audio processors.
- class [DeviceCrawlerIface](#)
IDL interface class for `Bse::DeviceCrawler`.
- class [DeviceCrawlerImpl](#)
- class [Devicelface](#)
IDL interface class for `Bse::Device`.
- class [DeviceImpl](#)
- struct [DeviceInfo](#)
Info for device types.
- struct [DriverEntry](#)
Driver information for PCM and MIDI handling.
- struct [DriverEntrySeq](#)
`DriverEntry` sequence.
- interface [EditableSample](#)
Interface for editable PCM wave samples.
- class [EditableSampleIface](#)
IDL interface class for `Bse::EditableSample`.
- class [EditableSampleImpl](#)
- class [Flac1Handle](#)
`Flac1Handle` supports storing flac files as binary appendix to BSE files.
- struct [FloatSeq](#)
A list of floating point values.
- class [FriendAllocator](#)

- *A `std::make_shared<>()` wrapper class to access private ctor & dtor.*
- struct `Icon`
 - *Representation of an icon pixel image.*
- interface `Item`
 - *Base interface type for objects that can be added to a container.*
- class `ItemIface`
 - *IDL interface class for `Bse::Item`.*
- class `ItemImpl`
- struct `ItemSeq`
 - *A list of `Item` or derived objects.*
- class `KeccakCryptoRng`
 - *`KeccakCryptoRng` - A KeccakF1600 based cryptographic quality pseudo-random number generator.*
- class `KeccakFastRng`
 - *`KeccakFastRng` - A KeccakF1600 based fast pseudo-random number generator.*
- class `KeccakGoodRng`
 - *`KeccakGoodRng` - A KeccakF1600 based good quality pseudo-random number generator.*
- class `KeccakRng`
 - *`KeccakRng` - A KeccakF1600 based pseudo-random number generator.*
- interface `LegacyObject`
 - *Base type for all legacy objects, derived from `struct BseObject`.*
- class `LegacyObjectIface`
 - *IDL interface class for `Bse::LegacyObject`.*
- class `LegacyObjectImpl`
- interface `MidiNotifier`
 - *Interface for MIDI event notification.*
- class `MidiNotifierIface`
 - *IDL interface class for `Bse::MidiNotifier`.*
- class `MidiNotifierImpl`
- interface `MidiSynth`
 - *Interface for MIDI synthesis networks.*
- class `MidiSynthIface`
 - *IDL interface class for `Bse::MidiSynth`.*
- class `MidiSynthImpl`
- class `Module`
 - *Interface for the encapsulation of audio processors.*
- class `ModuleIface`
 - *IDL interface class for `Bse::Module`.*
- class `ModuleImpl`
- struct `ModuleTypeInfo`
 - *Info for module types.*
- struct `NoteDescription`
 - *A note description provides all needed details about a specific note. ”.*
- interface `Object`
 - *Base type for all new style C++ objects.*
- class `ObjectIface`
 - *IDL interface class for `Bse::Object`.*
- class `ObjectImpl`
- interface `Part`
 - *Data interface for containment of piano notes and MIDI effects.*
- struct `PartControl`
 - *`Part` specific control event representation.*
- struct `PartControlSeq`

- A list of part control events.*
- class [PartIface](#)
 - IDL interface class for `Bse::Part`.*
- class [PartImpl](#)
- struct [PartLink](#)
 - Record representing the use of a `Part` within a `Track` at a specific position.*
- struct [PartLinkSeq](#)
 - Sequence of `PartLink` records.*
- struct [PartNote](#)
 - `Part` specific note event representation.*
- struct [PartNoteSeq](#)
 - A list of part note events.*
- struct [PartSeq](#)
 - A list of `Part` or derived types.*
- class [Pcg32Rng](#)
 - `Pcg32Rng` is a permutating linear congruential PRNG.*
- interface [PcmWriter](#)
 - Interface for writing PCM wave data.*
- class [PcmWriterIface](#)
 - IDL interface class for `Bse::PcmWriter`.*
- class [PcmWriterImpl](#)
- struct [PixelSeq](#)
 - Representation of an image pixel sequence in ARGB format.*
- struct [ProbeFeatures](#)
 - Bits representing a selection of probe sample data features.*
- interface [Project](#)
 - Projects support loading, saving, playback and act as containers for all other sound objects.*
- class [ProjectIface](#)
 - IDL interface class for `Bse::Project`.*
- class [ProjectImpl](#)
- struct [PropertyCandidates](#)
 - A list of items suitable to set as a specific property value.*
- class [Resampler2](#)
 - Interface for factor 2 resampling classes.*
- struct [SampleFileInfo](#)
 - Structure containing meta data for multi wave samples.*
- class [Sequencer](#)
 - Note and MIDI sequencer.*
- interface [Server](#)
 - Main Bse remote origin object.*
- class [ServerIface](#)
 - IDL interface class for `Bse::Server`.*
- class [ServerImpl](#)
- struct [SHA3_224](#)
 - `SHA3_224` - 224 Bit digest generation.*
- struct [SHA3_256](#)
 - `SHA3_256` - 256 Bit digest generation.*
- struct [SHA3_384](#)
 - `SHA3_384` - 384 Bit digest generation.*
- struct [SHA3_512](#)
 - `SHA3_512` - 512 Bit digest generation.*
- struct [SHAKE128](#)

- SHAKE128 - 128 Bit extendable output digest generation.*
- struct [SHAKE256](#)
 - SHAKE256 - 256 Bit extendable output digest generation.*
- struct [SharedMemory](#)
 - Descriptor for a shared memory region.*
- struct [ShmFragment](#)
 - Fragment description for interesting bits of shared memory.*
- struct [ShmFragmentSeq](#)
 - Collection of shared memory fragments.*
- interface [SignalMonitor](#)
 - Interface for monitoring output signals.*
- class [SignalMonitorIface](#)
 - IDL interface class for `Bse::SignalMonitor`.*
- class [SignalMonitorImpl](#)
- interface [SNet](#)
 - Base interface type for all kinds of synthesis networks.*
- class [SNetIface](#)
 - IDL interface class for `Bse::SNet`.*
- class [SNetImpl](#)
- interface [Song](#)
 - Interface for `Track` and `Part` objects, as well as meta data for sequencing.*
- class [SongIface](#)
 - IDL interface class for `Bse::Song`.*
- class [SongImpl](#)
- struct [SongTiming](#)
 - Song timing configuration.*
- interface [SoundFont](#)
 - Interface for sound fonts.*
- class [SoundFontIface](#)
 - IDL interface class for `Bse::SoundFont`.*
- class [SoundFontImpl](#)
- class [SoundFontRepoIface](#)
 - IDL interface class for `Bse::SoundFontRepo`.*
- class [SoundFontRepoImpl](#)
- interface [Source](#)
 - Base interface type for synthesis modules with input or output streams.*
- class [SourceIface](#)
 - IDL interface class for `Bse::Source`.*
- class [SourceImpl](#)
- class [Spinlock](#)
 - The `Spinlock` uses low-latency busy spinning to acquire locks.*
- class [Strings](#)
 - Convenience Constructor for `StringSeq` or `std::vector<std::string>`*
- struct [StringSeq](#)
 - Stringeq - a variable length list of test strings.*
- interface [SubSynth](#)
 - Synthesizer module for embedding (rerouting input and output) of another synthesizer network (`SNet`).*
- class [SubSynthIface](#)
 - IDL interface class for `Bse::SubSynth`.*
- class [SubSynthImpl](#)
- interface [Super](#)
 - Base interface type for `Item` managers.*

- class `SuperInterface`
IDL interface class for `Bse::Super`.
- class `SuperImpl`
- struct `SuperSeq`
A list of `Super` type objects.
- class `TaskRegistry`
The task registry keeps track of runtime threads for profiling and statistical purposes.
- struct `TaskStatus`
Acquire information about a task (process or thread) at runtime.
- interface `Track`
Interface for sequencing information and links to `Part` objects.
- class `TrackInterface`
IDL interface class for `Bse::Track`.
- class `TrackImpl`
- struct `TrackPart`
Structure linking to a `Track` from within a `Part`.
- struct `TrackPartSeq`
Sequence of `TrackPart` records.
- struct `TrackSeq`
Sequence of `Track` objects.
- struct `UserMessage`
Structure for submission of user interface messages from BSE.
- interface `Wave`
Interface for PCM wave samples.
- class `WaveInterface`
IDL interface class for `Bse::Wave`.
- class `WaveImpl`
- interface `WaveOsc`
Oscillator module for wave files.
- class `WaveOscInterface`
IDL interface class for `Bse::WaveOsc`.
- class `WaveOscImpl`
- struct `WaveOscSeq`
A list of part note events.
- interface `WaveRepo`
Interface serving as container for `Wave` objects.
- class `WaveRepoInterface`
IDL interface class for `Bse::WaveRepo`.
- class `WaveRepoImpl`

Typedefs

- typedef `uint8_t uint8`
An 8-bit unsigned integer.
- typedef `uint16_t uint16`
A 16-bit unsigned integer.
- typedef `uint32_t uint32`
A 32-bit unsigned integer.
- typedef `uint64_t uint64`
*A 64-bit unsigned integer, use `PRI*64` in format strings.*
- typedef `int8_t int8`
An 8-bit signed integer.

- typedef **int16_t** **int16**
A 16-bit signed integer.
- typedef **int32_t** **int32**
A 32-bit signed integer.
- typedef **int64_t** **int64**
A 64-bit unsigned integer, use PRI*64 in format strings.
- typedef **uint32_t** **unichar**
A 32-bit unsigned integer used for Unicode characters.
- typedef **std::string** **String**
Convenience alias for **std::string**.
- using **StringVector** = **vector**< **String** >
Convenience alias for a **std::vector**<**std::string**>.
- typedef **uint32_t** **uint**
Provide 'uint' as convenience type.

Enumerations

- enum **MonitorField** {
F64_GENERATION, F32_MIN, F32_MAX, F32_DB_SPL,
F32_DB_TIP, END_BYTE }
Offsets for signal monitoring fields in bytes, field type and size is used as prefix.
- enum **UserMessageType** { **ERROR**, **WARNING**, **INFO**, **DEBUG** }
- enum **SongTelemetry** { **I32_TICK_POINTER**, **BYTECOUNT** }
Offsets for signal monitoring fields in bytes, field type and size is used as prefix.
- enum **ProjectState** { **INACTIVE**, **ACTIVE**, **PLAYING** }
Enumeration describing the current activation and playback state of a project.
- enum **ModuleFlag** { **ModuleFlag::NORMAL**, **ModuleFlag::CHEAP**, **ModuleFlag::EXPENSIVE**, **ModuleFlag::VIRTUAL_** }

Functions

- bool **print_backtrace** (const **char** *file, **int** line, const **char** *func)
Print a C++ backtrace to stdout/stderr.
- **int** **fmsb** (uint64 val)
The **fmsb()** function returns the position of the most significant bit set in the word val.
- void * **aligned_alloc** (**size_t** total_size, **size_t** alignment, **uint8** **free_pointer)
Allocate a block of memory aligned to at least alignment bytes.
- void **aligned_free** (uint8 **free_pointer)
Release a block of memory allocated through **aligned_malloc()**.
- **String** **feature_toggle_find** (const **String** &config, const **String** &feature, const **String** &fallback)
Find feature in config, return its value or fallback.
- bool **feature_toggle_bool** (const **char** *config, const **char** *feature)
Check for feature in config, if feature is empty, checks for any feature.
- bool **feature_check** (const **char** *feature)
Check if feature is enabled via **\$BSE_FEATURE**.
- bool **url_show** (const **char** *url)
Find a suitable WWW user agent (taking user configurations into account) and start it to display url.
- bool **debug_key_enabled** (const **char** *conditional)
Check if *conditional* is enabled by **\$BSE_DEBUG**.
- bool **debug_key_enabled** (const :: **std::string** &conditional)
Check if *conditional* is enabled by **\$BSE_DEBUG**.
- :: **std::string** **debug_key_value** (const **char** *conditional)
Retrieve the value assigned to debug key *conditional* in **\$BSE_DEBUG**.

- void `diag_abort_hook` (const `std::function`< void(const :: `std::string` &)> &hook)
Call hook for fatal_error() and diag_failed_assert().
- template<class... Args>
`String` `string_format` (const `char` *format, const Args &...args)
Formatted printing ala printf() into a String, using the POSIX/C locale.
- template<class ... Args>
`String` `String` void `fatal_error` (const `char` *format, const Args &...args)
Issue a printf-like message and abort the program, this function will not return.
- template<class ... Args>
void `warning` (const `char` *format, const Args &...args)
Issue a printf-like warning message.
- template<class ... Args>
void `info` (const `char` *format, const Args &...args)
Issue an informative printf-like message.
- template<class... Args>
void `printout` (const `char` *format, const Args &...args)
Print a message on stdout (and flush stdout) ala printf(), using the POSIX/C locale.
- bool `debug_enabled` ()
Check if any kind of debugging is enabled by \$BSE_DEBUG.
- template<class ... Args>
void `debug` (const `char` *cond, const `char` *format, const Args &...args)
Issue a printf-like debugging message if cond is enabled by \$BSE_DEBUG.
- template<class ... Args>
void `debug_message` (const `char` *file, int line, const `char` *func, const `char` *cond, const `char` *format, const Args &...args)
Issue a printf-like debugging message if cond is enabled by \$BSE_DEBUG.
- template<class V >
bool `vector_erase_element` (V &v, const typename V::value_type &value)
Erase element value from std::vector v if it is present.
- template<class V , class O >
bool `vector_erase_iface` (V &v, O *value)
Erase element value from std::vector v if it matches a vector elements iface().
- template<class InputIterator , class OutputIterator >
OutputIterator `copy_reordered` (InputIterator const unordered_first, InputIterator const unordered_end, InputIterator const ordered_first, InputIterator const ordered_end, OutputIterator output_iterator)
Copy unordered_first .. unordered_end into output_iterator in the order given by ordered_first .. ordered_end.
- template<typename RandIter , class Cmp , typename Arg >
std::pair< RandIter, bool > `binary_lookup_insertion_pos` (RandIter begin, RandIter end, Cmp cmp, ← elements, const Arg &arg)
Perform a binary lookup to find the insertion position for a new element.
- template<typename RandIter , class Cmp , typename Arg >
RandIter `binary_lookup_sibling` (RandIter begin, RandIter end, Cmp cmp_elements, const Arg &arg)
Perform a binary lookup to yield exact or nearest match.
- template<typename RandIter , class Cmp , typename Arg >
RandIter `binary_lookup` (RandIter begin, RandIter end, Cmp cmp_elements, const Arg &arg)
Perform binary lookup and yield exact match or end.
- template<class ... Args>
void `fatal_error` (const `char` *format, const Args &...args)
Issue a printf-like message and abort the program, this function will not return.
- template<class... Args>
void `printerr` (const `char` *format, const Args &...args)
Print a message on stderr (and flush stderr) ala printf(), using the POSIX/C locale.
- void `zintern_free` (uint8 *dc_data)
Free data returned from zintern_decompress().

- [uint8 * zintern_decompress](#) (unsigned **int** decompressed_size, const unsigned **char** *cdata, unsigned **int** cdata_size)
Decompress data via zlib.
- [ServerHandle init_server_instance](#) ()
Retrieve a handle for the `Bse::Server` instance managing the `Bse` thread.
- [uint exec_now](#) (const **std::function**< bool()> &function)
Run function immediately with the next event loop iteration, return `true` to keep alive.
- [uint exec_now](#) (const **std::function**< void()> &function)
Run function immediately with the next event loop iteration.
- [uint exec_timeout](#) (const **std::function**< bool()> &function, **uint** delay_ms)
Run function after `delay_ms` milliseconds have passed, return `true` to keep alive.
- [uint exec_timeout](#) (const **std::function**< void()> &function, **uint** delay_ms)
Run function after `delay_ms` milliseconds have passed.
- [bool exec_handler_clear](#) (**uint** id)
Remove a function previously added with `exec_now()` or `exec_timeout()`
- [SfiGlueContext * init_glue_context](#) (const **gchar** *client, const **std::function**< void()> &caller_wakeup)
Create SFI glue layer context.
- [void init_async](#) (**int** *argc, **char** **argv, const **char** *app_name, const **StringVector** &args)
Initialize and start BSE.
- [bool init_needed](#) ()
Check whether `init_async()` still needs to be called.
- [Icon icon_from_pixstream](#) (const **uint8** *pixstream)
Create a `Bse::Icon` from a `GdkPixbuf` pixstream.
- [bool icon_sanitize](#) (**Icon** *icon)
Ensure consistency of the icon fields.
- [constexpr bool constexpr_equals](#) (const **char** *a, const **char** *b, **size_t** n)
Test string equality at compile time.
- [template<class Type, class ... Ts> void new_inplace](#) (Type &typemem, Ts &&... args)
Call inplace new operator by automatically inferring the Type.
- [template<class Type> void delete_inplace](#) (Type &typemem)
Call inplace delete operator by automatically inferring the Type.
- [template<class Target, class Source> std::shared_ptr< typename std::remove_pointer< Target >::type > shared_ptr_cast](#) (Source *object)
Shorthand for `std::dynamic_pointer_cast<>(shared_from_this())`.
- [template<class Target, class Source> const std::shared_ptr< typename std::remove_pointer< Target >::type > shared_ptr_cast](#) (const Source *object)
See `shared_ptr_cast(Source)`.*
- [template<class Target, class Source> std::shared_ptr< typename std::remove_pointer< Target >::type > shared_ptr_cast](#) (std::shared_ptr< Source > &sptr)
See `shared_ptr_cast(Source)`.*
- [template<class Target, class Source> const std::shared_ptr< typename std::remove_pointer< Target >::type > shared_ptr_cast](#) (const std::shared_ptr< Source > &sptr)
See `shared_ptr_cast(Source)`.*
- [void collect_runtime_entropy](#) (**uint64** *data, **size_t** n)
To provide good quality random number seeds, this function gathers entropy from a variety of process specific sources.
- [void collect_system_entropy](#) (**uint64** *data, **size_t** n)
This function adds to `collect_runtime_entropy()` by collecting entropy from additional but potentially slower system sources, such as interrupt counters, disk + network statistics, system load, execution + pipelining + scheduling latencies and device MACs.

- MemoryArea [find_memory_area](#) (uint32 mem_id)
Lookup a previously created memory area.
- MemoryArea [create_memory_area](#) (uint32 mem_size, uint32 alignment = BSE_CACHE_LINE_ALIGNME←
NT)
Create isolated memory area, the MemoryArea.mem_id can be used for [allocate_aligned_block\(\)](#).
- AlignedBlock [allocate_aligned_block](#) (uint32 mem_id, uint32 length)
Create a memory block from memory area mem_id, if 0, uses the internal cache-line aligned pool.
- void [release_aligned_block](#) (const AlignedBlock &block)
Release a previously allocated block.
- **std::string** [runpath](#) (RPath rpath)
Retrieve various resource paths at runtime.
- const **char** *() _ (const **char** * string)
Translate message strings in the BEAST/BSE text domain.
- **std::string**() _ (const **std::string** & string)
Translate message strings in the BEAST/BSE text domain.
- const **char** *() _ (const **char** * string, const **char** *plural, int64_t n)
Translate message strings in the BEAST/BSE text domain, use forms if != 1.
- **std::string**() _ (const **std::string** & string, const **std::string** &plural, int64_t n)
Translate message strings in the BEAST/BSE text domain, use forms if != 1.
- **std::string** [cpu_arch](#) ()
Retrieve string identifying the CPU architecture.
- **String** [cpu_info](#) ()
The returned string contains: number of online CPUs, a string describing the CPU architecture, the vendor and finally a number of flag words describing CPU features plus a trailing space.
- uint64 [timestamp_startup](#) ()
Provides the [timestamp_realtime\(\)](#) value from program startup.
- uint64 [timestamp_realtime](#) ()
Return the current time as uint64 in μseconds.
- uint64 [timestamp_resolution](#) ()
Provide resolution of [timestamp_benchmark\(\)](#) in nano-seconds.
- uint64 [timestamp_benchmark](#) ()
Returns benchmark timestamp in nano-seconds, clock starts around program startup.
- **String** [timestamp_format](#) (uint64 stamp, uint maxlen)
Convert stamp into a string, adding μsecond fractions if space permits.
- uint64 [monotonic_counter](#) ()
A monotonically increasing counter, increments are atomic and visible in all threads.
- **std::string** [executable_path](#) ()
Retrieve the path to the currently running executable.
- **std::string** [executable_name](#) ()
Retrieve the name part of [executable_path\(\)](#).
- **std::string** [version](#) ()
Provide a string containing the BSE library build.
- **String** [program_alias](#) ()
Retrieve the program name as used for logging or debug messages.
- void [program_alias_init](#) (**String** customname)
Set program_alias to a non-localized alias other than program_argv0 if desired.
- **String** [application_name](#) ()
Retrieve the localized program name intended for user display.
- void [application_name_init](#) (**String** desktopname)
Set the application_name to a name other than program_alias if desired.

- **String** `program_cwd` ()
The current working directory during startup.
- **void** `breakpoint` ()
Cause a debugging breakpoint, for development only.
- **void** `sha3_224_hash` (const void *data, **size_t** data_length, **uint8_t** hashvalue[28])
Calculate 224 bit SHA3 digest from data, see also class [SHA3_224](#).
- **void** `sha3_256_hash` (const void *data, **size_t** data_length, **uint8_t** hashvalue[32])
Calculate 256 bit SHA3 digest from data, see also class [SHA3_256](#).
- **void** `sha3_384_hash` (const void *data, **size_t** data_length, **uint8_t** hashvalue[48])
Calculate 384 bit SHA3 digest from data, see also class [SHA3_384](#).
- **void** `sha3_512_hash` (const void *data, **size_t** data_length, **uint8_t** hashvalue[64])
Calculate 512 bit SHA3 digest from data, see also class [SHA3_512](#).
- **void** `shake128_hash` (const void *data, **size_t** data_length, **uint8_t** *hashvalues, **size_t** n)
Calculate SHA3 extendable output digest for 128 bit security strength, see also class [SHAKE128](#).
- **void** `shake256_hash` (const void *data, **size_t** data_length, **uint8_t** *hashvalues, **size_t** n)
Calculate SHA3 extendable output digest for 256 bit security strength, see also class [SHAKE256](#).
- **uint64_t** `random_int64` ()
Generate a non-deterministic, uniformly distributed 64 bit pseudo-random number.
- **int64_t** `random_irange` (**int64_t** begin, **int64_t** end)
Generate uniformly distributed pseudo-random integer within range.
- **double** `random_float` ()
Generate uniformly distributed pseudo-random floating point number.
- **double** `random_frange` (**double** begin, **double** end)
Generate uniformly distributed pseudo-random floating point number within a range.
- **uint64_t** `random_nonce` ()
Provide a unique 64 bit identifier that is not 0, see also [random_int64\(\)](#).
- **void** `random_secret` (**uint64_t** *secret_var)
Generate a secret non-zero nonce in secret_var, unless it has already been assigned.
- **std::string** `beastbse_cachedir_create` ()
Create exclusive cache directory for this process' runtime.
- **std::string** `beastbse_cachedir_current` ()
Retrieve (or create) the temporary cache directory for this runtime.
- **void** `beastbse_cachedir_cleanup` ()
Clean stale cache directories from past runtimes, may be called from any thread.
- **String** `string_multiply` (const **String** &s, **uint64** count)
Reproduce a string s for count times.
- **String** `string_canonify` (const **String** &string, const **String** &valid_chars, const **String** &substitute)
Enforce a canonical charset for a string.
- **bool** `string_is_canonified` (const **String** &string, const **String** &valid_chars)
Check if `string_canonify()` would modify string.
- **String** `string_set_a2z` ()
Returns a string containing all of a-z.
- **String** `string_set_A2Z` ()
Returns a string containing all of A-Z.
- **String** `string_set_ascii_alnum` ()
Returns a string containing all of 0-9, A-Z and a-z.
- **String** `string_tolower` (const **String** &str)
Convert all string characters into Unicode lower case characters.
- **String** `string_toupper` (const **String** &str)
Convert all string characters into Unicode upper case characters.
- **String** `string_totitle` (const **String** &str)
Convert all string characters into Unicode title characters.

- **String** `string_capitalize` (const **String** &str, **size_t** maxn)
Capitalize words, so the first letter is upper case, the rest lower case.
- **String** `string_normalize_nfc` (const **String** &src)
Yield normalized composed UTF-8 string.
- **String** `string_normalize_nfd` (const **String** &src)
Yield normalized decomposed UTF-8 string.
- **String** `string_normalize_nfkc` (const **String** &src)
Formatting stripped normalized composed UTF-8 string.
- **String** `string_normalize_nfkd` (const **String** &src)
Formatting stripped normalized decomposed UTF-8 string.
- **String** `string_casefold` (const **String** &src)
Yield UTF-8 string useful for case insensitive comparisons.
- **int** `string_cmp` (const **String** &s1, const **String** &s2)
Like strcmp(3) for UTF-8 strings.
- **int** `string_casecmp` (const **String** &s1, const **String** &s2)
Like strcasecmp(3) for UTF-8 strings.
- **String** `string_vprintf` (const **char** *format, **va_list** vargs)
Formatted printing ala vprintf() into a String, using the POSIX/C locale.
- **String** `string_locale_vprintf` (const **char** *format, **va_list** vargs)
Formatted printing like string_vprintf using the current locale.
- **StringVector** `string_split` (const **String** &string, const **String** &splitter, **size_t** maxn)
Split a string, using splitter as delimiter.
- **StringVector** `string_split_any` (const **String** &string, const **String** &splitchars, **size_t** maxn)
Split a string, using any of the splitchars as delimiter.
- **void** `string_vector_erase_empty` (**StringVector** &svector)
Remove empty elements from a string vector.
- **void** `string_vector_lstrip` (**StringVector** &svector)
Left-strip all elements of a string vector, see string_lstrip().
- **void** `string_vector_rstrip` (**StringVector** &svector)
Right-strip all elements of a string vector, see string_rstrip().
- **void** `string_vector_strip` (**StringVector** &svector)
Strip all elements of a string vector, see string_strip().
- **String** `string_join` (const **String** &junctor, const **StringVector** &strvec)
Join a number of strings.
- **bool** `string_to_bool` (const **String** &string, **bool** fallback)
Interpret a string as boolean value.
- **String** `string_from_bool` (**bool** value)
Convert a boolean value into a string.
- **uint64** `string_to_uint` (const **String** &string, **size_t** *consumed, **uint** base)
Parse a string into a 64bit unsigned integer, optionally specifying the expected number base.
- **String** `string_from_uint` (**uint64** value)
Convert a 64bit unsigned integer into a string.
- **bool** `string_has_int` (const **String** &string)
Checks if a string contains a digit, optionally preceeded by whitespaces.
- **int64** `string_to_int` (const **String** &string, **size_t** *consumed, **uint** base)
Parse a string into a 64bit integer, optionally specifying the expected number base.
- **String** `string_from_int` (**int64** value)
Convert a 64bit signed integer into a string.
- **long double** `posix_locale_strtold` (const **char** *nptr, **char** **endptr)
Parse a double from a string ala strtod(), trying locale specific characters and POSIX/C formatting.
- **long double** `current_locale_strtold` (const **char** *nptr, **char** **endptr)
Parse a double from a string ala strtod(), trying locale specific characters and POSIX/C formatting.

- **double** `string_to_double` (const **String** & **string**)
Parse a double from a string, trying locale specific characters and POSIX/C formatting.
- **double** `string_to_double` (const **char** *dblstring, const **char** **endptr)
Similar to `string_to_double(const String&)`, but returns the first failing character position in `endptr`.
- **String** `string_from_float` (**float** value)
Convert a float into a string, using the POSIX/C locale.
- **String** `string_from_double` (**double** value)
Convert a double into a string, using the POSIX/C locale.
- **vector**< **double** > `string_to_double_vector` (const **String** & **string**)
Parse a string into a list of doubles, expects ';' as delimiter.
- **String** `string_from_double_vector` (const **vector**< **double** > &dvec, const **String** &delim)
Construct a string out of all double values passed in `dvec`, separated by `delim`.
- **String** `string_from_errno` (**int** `errno_val`)
Returns a `String` describing the passed in `errno` value, similar to `strerror()`.
- **bool** `string_is_uuid` (const **String** &uuid_string)
Returns whether `uuid_string` contains a properly formatted UUID string.
- **int** `string_cmp_uuid` (const **String** &uuid_string1, const **String** &uuid_string2)
Returns whether `uuid_string1` compares smaller (-1), equal (0) or greater (+1) to `uuid_string2`.
- **bool** `string_startswith` (const **String** & **string**, const **String** &fragment)
Returns whether `string` starts with `fragment`.
- **bool** `string_endswith` (const **String** & **string**, const **String** &fragment)
Returns whether `string` ends with `fragment`.
- **bool** `string_match_identifier_tail` (const **String** &ident, const **String** &tail)
Variant of `string_match_identifier()` that matches tail against `ident` at word boundary.
- **bool** `string_match_identifier` (const **String** &ident1, const **String** &ident2)
Check equality of strings canonicalized to "[0-9a-z]+".
- **String** `string_from_pretty_function_name` (const **char** *cxx_pretty_function)
Extract the full function name from `PRETTY_FUNCTION`.
- **String** `string_to_escaped` (const **String** &str)
Escape text like a C string.
- **String** `string_to_cquote` (const **String** &str)
Returns a string as C string including double quotes.
- **String** `string_from_cquote` (const **String** &input)
Parse a possibly quoted C string into regular string.
- **String** `string_lstrip` (const **String** &input)
Strip whitespaces from the left of a string.
- **String** `string_rstrip` (const **String** &input)
Strip whitespaces from the right of a string.
- **String** `string_strip` (const **String** &input)
Strip whitespaces from the left and right of a string.
- **String** `string_replace` (const **String** &input, const **String** &marker, const **String** &replacement, **size_t** `maxn`)
Replace substring `marker` in `input` with `replacement`, at most `maxn` times.
- **String** `string_substitute_char` (const **String** &input, const **char** `match`, const **char** `subst`)
Replace all occurrences of `match` in `input` with `subst`.
- **String** `string_hexdump` (const **void** *addr, **size_t** `length`, **size_t** `initial_offset`)
Produce hexdump of a memory region.
- **void** `memset4` (**uint32** *mem, **uint32** `filler`, **uint** `length`)
Fill a memory area with a 32-bit quantity.
- **String** `string_vector_find` (const **StringVector** &svector, const **String** &prefix, const **String** &fallback)
Search for `prefix` in `svector` and return the matching element.
- **String** `string_vector_find_value` (const **StringVector** &svector, const **String** &prefix, const **String** &fallback)
Search for `prefix` in `svector` and return remainder of the matching string.

- **StringVector** `cstrings_to_vector` (const **char** *s,...)

Construct a StringVector from a NULL terminated list of string arguments.
- void **string_options_split** (const **String** &option_string, **vector**< **String** > &option_names, **vector**< **String** > &option_values, const **String** &empty_default)

Split an option list string into name/value pairs.
- **String** **string_option_get** (const **String** &option_string, const **String** &option)

Retrieve the option value from an options list separated by ':' or ';'.
- bool **string_option_check** (const **String** &option_string, const **String** &option)

Check if an option is set/unset in an options list string.
- bool **text_convert** (const **String** &to_charset, **String** &output_string, const **String** &from_charset, const **String** &input_string, const **String** &fallback_charset, const **String** &output_mark)

Convert a string from one encoding to another.
- template<typename Type >
Type **string_to_type** (const **String** &string)

Convert a string to template argument type, such as bool, int, double.
- template<typename Type >
String **string_from_type** (Type value)

Create a string from a templated argument value, such as bool, int, double.
- template<class... Args>
String **string_locale_format** (const **char** *format, const Args &...args)

Formatted printing ala printf() into a String, using the current locale.
- **size_t** **utf8len** (const **char** *str)

Count valid UTF-8 sequences, invalid sequences are counted as Latin-1 characters.
- **size_t** **utf8len** (const **std::string** &str)

Count valid UTF-8 sequences, invalid sequences are counted as Latin-1 characters.
- **size_t** **utf8_to_unicode** (const **char** *str, **uint32_t** *codepoints)

Convert valid UTF-8 sequences to Unicode codepoints, invalid sequences are treated as Latin-1 characters.
- **size_t** **utf8_to_unicode** (const **std::string** &str, **std::vector**< **uint32_t** > &codepoints)

Convert valid UTF-8 sequences to Unicode codepoints, invalid sequences are treated as Latin-1 characters.
- **std::string** **string_from_unicode** (const **uint32_t** *codepoints, **size_t** n_codepoints)

Convert codepoints into an UTF-8 string, using the shortest possible encoding.
- **std::string** **string_from_unicode** (const **std::vector**< **uint32_t** > &codepoints)

Convert codepoints into an UTF-8 string, using the shortest possible encoding.
- constexpr bool **unicode_is_valid** (**uint32_t** u)

Return whether u is an allowed Unicode codepoint within 0x10FFFF and not part of a UTF-16 surrogate pair.
- constexpr bool **unicode_is_assigned** (**uint32_t** u)

Return whether u matches any of the assigned Unicode planes.
- constexpr bool **unicode_is_noncharacter** (**uint32_t** u)

Return whether u is one of the 66 Unicode noncharacters.
- constexpr bool **unicode_is_character** (**uint32_t** u)

Return whether u is not one of the 66 Unicode noncharacters.
- constexpr bool **unicode_is_control_code** (**uint32_t** u)

Return whether u is one of the 65 Unicode control codes.
- constexpr bool **unicode_is_private** (**uint32_t** u)

Return whether u is in one of the 3 private use areas of Unicode.

Variables

- Const `KAMMER_NOTE`
Value represents unparsable/unknown notes.
- Const `KAMMER_FREQ`
Kammer note, representing the kammer frequency's MIDI note value for A' or A4.
- Const `KAMMER_OCTAVE`
Pitch Standard, see also: [https://en.wikipedia.org/wiki/A440_\(pitch_standard\)](https://en.wikipedia.org/wiki/A440_(pitch_standard))
- Const `MIN_OCTAVE`
Octave number for MIDI A'.
- Const `MAX_OCTAVE`
Octave of MIN_NOTE.
- Const `MIN_FINE_TUNE`
Octave of MAX_NOTE.
- `uint64_t` `cached_hash_secret`
Use `hash_secret()` for access.

Detailed Description

The `Bse` namespace contains all functions of the synthesis engine.

Typedef Documentation

`int16`

```
typedef int16_t Bse::int16
```

A 16-bit signed integer.

`int32`

```
typedef int32_t Bse::int32
```

A 32-bit signed integer.

`int64`

```
typedef int64_t Bse::int64
```

A 64-bit unsigned integer, use `PRI*64` in format strings.

`int8`

```
typedef int8_t Bse::int8
```

An 8-bit signed integer.

String

```
typedef std::string Bse::String
```

Convenience alias for `std::string`.

StringVector

```
typedef vector< String > Bse::StringVector
```

Convenience alias for a `std::vector<std::string>`.

uint

```
typedef uint32_t Bse::uint
```

Provide 'uint' as convenience type.

uint16

```
typedef uint16_t Bse::uint16
```

A 16-bit unsigned integer.

uint32

```
typedef uint32_t Bse::uint32
```

A 32-bit unsigned integer.

uint64

```
typedef uint64_t Bse::uint64
```

A 64-bit unsigned integer, use `PRI*64` in format strings.

uint8

```
typedef uint8_t Bse::uint8
```

An 8-bit unsigned integer.

unichar

```
typedef uint32_t Bse::unichar
```

A 32-bit unsigned integer used for Unicode characters.

Enumeration Type Documentation

ModuleFlag

```
enum Bse::ModuleFlag [strong]
```

Enumerator

NORMAL	Nutral flag.
CHEAP	Very short or NOP as process() function.
EXPENSIVE	Indicate lengthy process() functio.
VIRTUAL_	Flag used internally.

MonitorField

enum `Bse::MonitorField`

Offsets for signal monitoring fields in bytes, field type and size is used as prefix.

Enumerator

F64_GENERATION	Generation counter, updated on every modification.
F32_MIN	Minimum value of the last frame.
F32_MAX	Maximum value of the last frame.
F32_DB_SPL	Sound pressure level in dB SPL of the last frame.
F32_DB_TIP	Maximum recent dB SPL.
END_BYTE	Total length of all MonitorField values in bytes.

ProjectState

enum `Bse::ProjectState`

Enumeration describing the current activation and playback state of a project.

Enumerator

INACTIVE	The project is not yet hooked to the sound engine.
ACTIVE	The sound engine is activated (running) for this project.
PLAYING	The project is active and the sequencer is running.

SongTelemetry

enum `Bse::SongTelemetry`

Offsets for signal monitoring fields in bytes, field type and size is used as prefix.

Enumerator

I32_TICK_POINTER	Current song position pointer.
BYTECOUNT	Total length of all fields.

UserMessageType

enum `Bse::UserMessageType`

Enumerator

ERROR	Indicate a message about an error condition.
WARNING	Indicate a message about a possibly harmful condition.
INFO	Indicate an informational message.
DEBUG	Indicate a debugging message (usually insignificant).

Function Documentation

`_O` [1/4]

```
const char*() Bse::_ (
    const char * string )
```

Translate message strings in the BEAST/BSE text domain.

`_O` [2/4]

```
std::string() Bse::_ (
    const std::string & string )
```

Translate message strings in the BEAST/BSE text domain.

`_O` [3/4]

```
const char*() Bse::_ (
    const char * string,
    const char * plural,
    int64_t n )
```

Translate message strings in the BEAST/BSE text domain, use forms if `!= 1`.

`_O` [4/4]

```
std::string() Bse::_ (
    const std::string & string,
    const std::string & plural,
    int64_t n )
```

Translate message strings in the BEAST/BSE text domain, use forms if `!= 1`.

`aligned_alloc()`

```
void * Bse::aligned_alloc (
    size_t total_size,
    size_t alignment,
    uint8 ** free_pointer )
```

Allocate a block of memory aligned to at least *alignment* bytes.

`aligned_free()`

```
void Bse::aligned_free (
    uint8 ** free_pointer )
```

Release a block of memory allocated through `aligned_malloc()`.

`allocate_aligned_block()`

```
AlignedBlock Bse::allocate_aligned_block (
    uint32 mem_id,
    uint32 length )
```

Create a memory block from memory area *mem_id*, if 0, uses the internal cache-line aligned pool.

application_name()

```
String Bse::application_name ( )
```

Retrieve the localized program name intended for user display.

application_name_init()

```
void Bse::application_name_init (
    String desktopname )
```

Set the application_name to a name other than program_alias if desired.

beastbse_cachedir_cleanup()

```
void Bse::beastbse_cachedir_cleanup ( )
```

Clean stale cache directories from past runtimes, may be called from any thread.

beastbse_cachedir_create()

```
std::string Bse::beastbse_cachedir_create ( )
```

Create exclusive cache directory for this process' runtime.

beastbse_cachedir_current()

```
std::string Bse::beastbse_cachedir_current ( )
```

Retrieve (or create) the temporary cache directory for this runtime.

binary_lookup()

```
template<typename RandIter , class Cmp , typename Arg >
```

```
RandIter Bse::binary_lookup (
    RandIter begin,
    RandIter end,
    Cmp cmp_elements,
    const Arg & arg ) [inline]
```

Perform binary lookup and yield exact match or *end*.

The arguments [*begin*, *end*] denote the range used for the lookup, *arg* is passed along with the current element to the *cmp_elements* function.

binary_lookup_insertion_pos()

```
template<typename RandIter , class Cmp , typename Arg >
std::pair<RandIter,bool> Bse::binary_lookup_insertion_pos (
    RandIter begin,
    RandIter end,
    Cmp cmp_elements,
    const Arg & arg ) [inline]
```

Perform a binary lookup to find the insertion position for a new element.

Return (end,false) for end-begin = 0, or return (position,true) for exact match, otherwise return (position,false) where position indicates the location for the key to be inserted (and may equal end).

binary_lookup_sibling()

```
template<typename RandIter , class Cmp , typename Arg >
RandIter Bse::binary_lookup_sibling (
    RandIter begin,
    RandIter end,
    Cmp cmp_elements,
    const Arg & arg ) [inline]
```

Perform a binary lookup to yield exact or nearest match.

return end for end-begin == 0, otherwise return the exact match element, or, if there's no such element, return the element last visited, which is pretty close to an exact match (will be one off into either direction).

breakpoint()

```
void Bse::breakpoint ( ) [inline]
```

Cause a debugging breakpoint, for development only.

collect_runtime_entropy()

```
void Bse::collect_runtime_entropy (
    uint64 * data,
    size_t n )
```

To provide good quality random number seeds, this function gathers entropy from a variety of process specific sources.

Collect entropy from the current process, usually quicker than [collect_system_entropy\(\)](#).

Under Linux, this includes the CPU counters, clocks and random devices. In combination with well established techniques like syscall timings (see Entropics13 [Swanson \(2013\)](#)) and a SHA3 algorithm derived random number generator for the mixing, the entropy collection is designed to be fast and good enough for all non-cryptographic uses. On an Intel Core i7, this function takes around 25µs.

collect_system_entropy()

```
void Bse::collect_system_entropy (
    uint64 * data,
    size_t n )
```

This function adds to [collect_runtime_entropy\(\)](#) by collecting entropy from additional but potentially slower system sources, such as interrupt counters, disk + network statistics, system load, execution + pipelining + scheduling latencies and device MACs.

Collect entropy from system devices, like interrupt counters, clocks and random devices.

The function is designed to yield random number seeds good enough to generate cryptographic tokens like session keys. On an Intel Core i7, this function takes around 2ms, so it's roughly 80 times slower than [collect_runtime_entropy\(\)](#).

constexpr_equals()

```
constexpr bool Bse::constexpr_equals (
    const char * a,
    const char * b,
    size_t n ) [inline]
```

Test string equality at compile time.

copy_reordered()

```
template<class InputIterator , class OutputIterator >
OutputIterator Bse::copy_reordered (
    InputIterator const unordered_first,
    InputIterator const unordered_end,
    InputIterator const ordered_first,
```

```
InputIterator const ordered_end,
OutputIterator output_iterator )
```

Copy *unordered_first .. unordered_end* into *output_iterator* in the order given by *ordered_first .. ordered_end*.

cpu_arch()

```
std::string Bse::cpu_arch ( )
```

Retrieve string identifying the CPU architecture.

cpu_info()

```
std::string Bse::cpu_info ( )
```

The returned string contains: number of online CPUs, a string describing the CPU architecture, the vendor and finally a number of flag words describing CPU features plus a trailing space.

Retrieve string identifying the runtime CPU type.

This allows checks for CPU features via a simple string search for "FEATURE".

Returns

```
Example: "4 AMD64 GenuineIntel FPU TSC HTT CMPXCHG16B MMX MMXEXT SSESYS SSE SSE2 SSE3
SSSE3 SSE4.1 SSE4.2 "
```

create_memory_area()

```
MemoryArea Bse::create_memory_area (
    uint32 mem_size,
    uint32 alignment )
```

Create isolated memory area, the `MemoryArea.mem_id` can be used for [allocate_aligned_block\(\)](#).

cstrings_to_vector()

```
StringVector Bse::cstrings_to_vector (
    const char * s,
    ... )
```

Construct a `StringVector` from a NULL terminated list of string arguments.

current_locale_strtold()

```
long double Bse::current_locale_strtold (
    const char * nptr,
    char ** endptr )
```

Parse a double from a string ala [strtod\(\)](#), trying locale specific characters and POSIX/C formatting.

debug()

```
template<class ... Args>
void Bse::debug (
    const char * cond,
    const char * format,
    const Args &... args ) [inline]
```

Issue a printf-like debugging message if `cond` is enabled by `$BSE_DEBUG`.

debug_enabled()

```
bool Bse::debug_enabled ( ) [inline]
```

Check if any kind of debugging is enabled by \$BSE_DEBUG.

debug_key_enabled() [1/2]

```
bool Bse::debug_key_enabled (
    const char * conditional )
```

Check if conditional is enabled by \$BSE_DEBUG.

debug_key_enabled() [2/2]

```
bool Bse::debug_key_enabled (
    const :: std::string & conditional )
```

Check if conditional is enabled by \$BSE_DEBUG.

debug_key_value()

```
std::string Bse::debug_key_value (
    const char * conditional )
```

Retrieve the value assigned to debug key conditional in \$BSE_DEBUG.

debug_message()

```
template<class ... Args>
void Bse::debug_message (
    const char * file,
    int line,
    const char * func,
    const char * cond,
    const char * format,
    const Args &... args ) [inline]
```

Issue a printf-like debugging message if cond is enabled by \$BSE_DEBUG.

delete_inplace()

```
template<class Type >
void Bse::delete_inplace (
    Type & typemem ) [inline]
```

Call inplace delete operator by automatically inferring the Type.

diag_abort_hook()

```
void Bse::diag_abort_hook (
    const std::function< void(const :: std::string &)> & hook )
```

Call hook for [fatal_error\(\)](#) and [diag_failed_assert\(\)](#).

exec_handler_clear()

```
bool Bse::exec_handler_clear (
    uint id )
```

Remove a function previously added with [exec_now\(\)](#) or [exec_timeout\(\)](#)

exec_now() [1/2]

```
uint Bse::exec_now (
    const std::function< bool()> & function )
```

Run function immediately with the next event loop iteration, return true to keep alive.

exec_now() [2/2]

```
uint Bse::exec_now (
    const std::function< void()> & function )
```

Run function immediately with the next event loop iteration.

exec_timeout() [1/2]

```
uint Bse::exec_timeout (
    const std::function< bool()> & function,
    uint delay_ms )
```

Run function after `delay_ms` milliseconds have passed, return true to keep alive.

exec_timeout() [2/2]

```
uint Bse::exec_timeout (
    const std::function< void()> & function,
    uint delay_ms )
```

Run function after `delay_ms` milliseconds have passed.

executable_name()

```
std::string Bse::executable_name ( )
```

Retrieve the name part of [executable_path\(\)](#).

executable_path()

```
std::string Bse::executable_path ( )
```

Retrieve the path to the currently running executable.

fatal_error() [1/2]

```
template<class ... Args>
String String void Bse::fatal_error (
    const char * format,
    const Args &... args )
```

Issue a printf-like message and abort the program, this function will not return.

Avoid using this in library code, aborting may take precious user data with it, library code should instead use [warning\(\)](#), [info\(\)](#) or [assert_return\(\)](#).

fatal_error() [2/2]

```
template<class ... Args>
void Bse::fatal_error (
    const char * format,
    const Args &... args )
```

Issue a printf-like message and abort the program, this function will not return.

Avoid using this in library code, aborting may take precious user data with it, library code should instead use [warning\(\)](#), [info\(\)](#) or [assert_return\(\)](#).

feature_check()

```
bool Bse::feature_check (
    const char * feature )
```

Check if feature is enabled via \$BSE_FEATURE.

feature_toggle_bool()

```
bool Bse::feature_toggle_bool (
    const char * config,
    const char * feature )
```

Check for *feature* in *config*, if *feature* is empty, checks for *any* feature.

feature_toggle_find()

```
String Bse::feature_toggle_find (
    const String & config,
    const String & feature,
    const String & fallback )
```

Find *feature* in *config*, return its value or *fallback*.

find_memory_area()

```
MemoryArea Bse::find_memory_area (
    uint32 mem_id )
```

Lookup a previously created memory area.

fmsb()

```
int Bse::fmsb (
    uint64 val )
```

The `fmsb()` function returns the position of the most significant bit set in the word *val*.

Find most significant bit set in a word.

The least significant bit is position 1 and the most significant position is, for example, 32 or 64.

Returns

The position of the most significant bit set is returned, or 0 if no bits were set.

icon_from_pixstream()

```
Icon Bse::icon_from_pixstream (
    const uint8 * pixstream )
```

Create a `Bse::Icon` from a GdkPixbuf pixstream.

icon_sanitize()

```
bool Bse::icon_sanitize (
    Icon * icon )
```

Ensure consistency of the *icon* fields.

info()

```
template<class ... Args>
void Bse::info (
    const char * format,
    const Args &... args )
```

Issue an informative printf-like message.

init_async()

```
void Bse::init_async (
    int * argc,
    char ** argv,
    const char * app_name,
    const StringVector & args )
```

Initialize and start BSE.

Initialize the BSE library and start the main BSE thread. Arguments specific to BSE are removed from *argc* / *argv*.

init_glue_context()

```
SfiGlueContext * Bse::init_glue_context (
    const gchar * client,
    const std::function< void()> & caller_wakeup )
```

Create SFI glue layer context.

Create and push an SFI glue layer context for the calling thread, to enable communications with the main BSE thread library.

init_needed()

```
bool Bse::init_needed ( )
```

Check wether [init_async\(\)](#) still needs to be called.

init_server_instance()

```
ServerHandle Bse::init_server_instance ( )
```

Retrieve a handle for the [Bse::Server](#) instance managing the [Bse](#) thread.

memset4()

```
void Bse::memset4 (
    uint32 * mem,
    uint32 filler,
    uint length )
```

Fill a memory area with a 32-bit quantity.

monotonic_counter()

```
uint64 Bse::monotonic_counter ( )
```

A monotonically increasing counter, increments are atomic and visible in all threads.

new_inplace()

```
template<class Type , class ... Ts>
void Bse::new_inplace (
    Type & typemem,
    Ts &&... args ) [inline]
```

Call inplace new operator by automatically inferring the Type.

posix_locale_strtold()

```
long double Bse::posix_locale_strtold (
    const char * nptr,
    char ** endptr )
```

Parse a double from a string ala **strtod()**, trying locale specific characters and POSIX/C formatting.

print_backtrace()

```
bool Bse::print_backtrace (
    const char * file,
    int line,
    const char * func ) [inline]
```

Print a C++ backtrace to stdout/stderr.

printerr()

```
template<class... Args>
void Bse::printerr (
    const char * format,
    const Args &... args )
```

Print a message on stderr (and flush stderr) ala **printf()**, using the POSIX/C locale.

printout()

```
template<class... Args>
void Bse::printout (
    const char * format,
    const Args &... args )
```

Print a message on stdout (and flush stdout) ala **printf()**, using the POSIX/C locale.

program_alias()

```
String Bse::program_alias ( )
```

Retrieve the program name as used for logging or debug messages.

program_alias_init()

```
void Bse::program_alias_init (
    String customname )
```

Set program_alias to a non-localized alias other than program_argv0 if desired.

program_cwd()

```
String Bse::program_cwd ( )
```

The current working directory during startup.

random_float()

```
double Bse::random_float ( )
```

Generate uniformly distributed pseudo-random floating point number.

This function generates a pseudo-random number like [random_int64\(\)](#), constrained to the range: $0.0 \leq \text{number} < 1.0$.

random_frange()

```
double Bse::random_frange (
    double begin,
    double end )
```

Generate uniformly distributed pseudo-random floating point number within a range.

This function generates a pseudo-random number like [random_float\(\)](#), constrained to the range: $\text{begin} \leq \text{number} < \text{end}$.

random_int64()

```
uint64_t Bse::random_int64 ( )
```

Generate a non-deterministic, uniformly distributed 64 bit pseudo-random number.

This function generates pseudo-random numbers using the system state as entropy and class [KeccakRng](#) for the mixing. No seeding is required.

random_irange()

```
int64_t Bse::random_irange (
    int64_t begin,
    int64_t end )
```

Generate uniformly distributed pseudo-random integer within range.

This function generates a pseudo-random number like [random_int64\(\)](#), constrained to the range: $\text{begin} \leq \text{number} < \text{end}$.

random_nonce()

```
uint64_t Bse::random_nonce ( )
```

Provide a unique 64 bit identifier that is not 0, see also [random_int64\(\)](#).

random_secret()

```
void Bse::random_secret (
    uint64_t * secret_var )
```

Generate a secret non-zero nonce in `secret_var`, unless it has already been assigned.

release_aligned_block()

```
void Bse::release_aligned_block (
    const AlignedBlock & am )
```

Release a previously allocated block.

runpath()

```
std::string Bse::runpath (
    RPath rpath )
```

Retrieve various resource paths at runtime.

sha3_224_hash()

```
void Bse::sha3_224_hash (
    const void * data,
    size_t data_length,
    uint8_t hashvalue[28] )
```

Calculate 224 bit SHA3 digest from *data*, see also class [SHA3_224](#).

sha3_256_hash()

```
void Bse::sha3_256_hash (
    const void * data,
    size_t data_length,
    uint8_t hashvalue[32] )
```

Calculate 256 bit SHA3 digest from *data*, see also class [SHA3_256](#).

sha3_384_hash()

```
void Bse::sha3_384_hash (
    const void * data,
    size_t data_length,
    uint8_t hashvalue[48] )
```

Calculate 384 bit SHA3 digest from *data*, see also class [SHA3_384](#).

sha3_512_hash()

```
void Bse::sha3_512_hash (
    const void * data,
    size_t data_length,
    uint8_t hashvalue[64] )
```

Calculate 512 bit SHA3 digest from *data*, see also class [SHA3_512](#).

shake128_hash()

```
void Bse::shake128_hash (
    const void * data,
    size_t data_length,
    uint8_t * hashvalues,
    size_t n )
```

Calculate SHA3 extendable output digest for 128 bit security strength, see also class [SHAKE128](#).

shake256_hash()

```
void Bse::shake256_hash (
    const void * data,
    size_t data_length,
    uint8_t * hashvalues,
    size_t n )
```

Calculate SHA3 extendable output digest for 256 bit security strength, see also class [SHAKE256](#).

`shared_ptr_cast()` [1/4]

```
template<class Target , class Source >
std::shared_ptr<typename std::remove_pointer<Target>::type> Bse::shared_ptr_cast (
    Source * object )
```

Shorthand for `std::dynamic_pointer_cast<>(shared_from_this())`.

A `shared_ptr_cast()` takes a `std::shared_ptr` or a pointer to an *object* that supports `std::enable_shared_from_this::shared_from_this()`. Using `std::dynamic_pointer_cast()`, the `shared_ptr` passed in (or retrieved via calling `shared_from_this()`) is cast into a `std::shared_ptr<Target>`, possibly resulting in an empty (NULL) `std::shared_ptr` if the underlying `dynamic_cast()` was not successful or if a NULL *object* was passed in. Note that `shared_from_this()` can throw a `std::bad_weak_ptr` exception if the object has no associated `std::shared_ptr` (usually during ctor and dtor), in which case the exception will also be thrown from `shared_ptr_cast<Target>()`. However a `shared_ptr_cast<Target*>()` call will not throw and yield an empty (NULL) `std::shared_ptr<Target>`. This is analogous to `dynamic_cast<T&>` which throws, versus `dynamic_cast<T*>` which yields NULL.

Returns

A `std::shared_ptr<Target>` storing a pointer to *object* or NULL.

Exceptions

<code>std::bad_weak_ptr</code>	if <code>shared_from_this()</code> throws, unless the <i>Target*</i> form is used.
--------------------------------	--

`shared_ptr_cast()` [2/4]

```
template<class Target , class Source >
const std::shared_ptr<typename std::remove_pointer<Target>::type> Bse::shared_ptr_cast (
    const Source * object )
```

See [shared_ptr_cast\(Source*\)](#).

`shared_ptr_cast()` [3/4]

```
template<class Target , class Source >
std::shared_ptr<typename std::remove_pointer<Target>::type> Bse::shared_ptr_cast (
    std::shared_ptr< Source > & sptr )
```

See [shared_ptr_cast\(Source*\)](#).

`shared_ptr_cast()` [4/4]

```
template<class Target , class Source >
const std::shared_ptr<typename std::remove_pointer<Target>::type> Bse::shared_ptr_cast (
    const std::shared_ptr< Source > & sptr )
```

See [shared_ptr_cast\(Source*\)](#).

`string_canonify()`

```
String Bse::string_canonify (
    const String & string,
    const String & valid_chars,
    const String & substitute )
```

Enforce a canonical charset for a string.

Convert all chars in *string* that are not listed as *valid_chars* with *substitute*.

string_capitalize()

```
String Bse::string_capitalize (
    const String & str,
    size_t maxn )
```

Capitalize words, so the first letter is upper case, the rest lower case.

string_casecmp()

```
int Bse::string_casecmp (
    const String & s1,
    const String & s2 )
```

Like `strcascmp(3)` for UTF-8 strings.

string_casefold()

```
String Bse::string_casefold (
    const String & src )
```

Yield UTF-8 string useful for case insensitive comparisons.

string_cmp()

```
int Bse::string_cmp (
    const String & s1,
    const String & s2 )
```

Like `strcmp(3)` for UTF-8 strings.

string_cmp_uuid()

```
int Bse::string_cmp_uuid (
    const String & uuid_string1,
    const String & uuid_string2 )
```

Returns whether *uuid_string1* compares smaller (-1), equal (0) or greater (+1) to *uuid_string2*.

string_endswith()

```
bool Bse::string_endswith (
    const String & string,
    const String & fragment )
```

Returns whether *string* ends with *fragment*.

string_format()

```
template<class... Args>
String Bse::string_format (
    const char * format,
    const Args &... args )
```

Formatted printing ala `printf()` into a String, using the POSIX/C locale.

string_from_bool()

```
String Bse::string_from_bool (
    bool value )
```

Convert a boolean value into a string.

string_from_cquote()

```
String Bse::string_from_cquote (
    const String & input )
```

Parse a possibly quoted C string into regular string.

string_from_double()

```
String Bse::string_from_double (
    double value )
```

Convert a double into a string, using the POSIX/C locale.

string_from_double_vector()

```
String Bse::string_from_double_vector (
    const vector< double > & dvec,
    const String & delim )
```

Construct a string out of all double values passed in *dvec*, separated by *delim*.

string_from_errno()

```
String Bse::string_from_errno (
    int errno_val )
```

Returns a String describing the passed in errno value, similar to **strerror()**.

string_from_float()

```
String Bse::string_from_float (
    float value )
```

Convert a float into a string, using the POSIX/C locale.

string_from_int()

```
String Bse::string_from_int (
    int64 value )
```

Convert a 64bit signed integer into a string.

string_from_pretty_function_name()

```
String Bse::string_from_pretty_function_name (
    const char * cxx_pretty_function )
```

Extract the full function name from **PRETTY_FUNCTION**.
See also **BSE_SIMPLE_FUNCTION**.

string_from_type()

```
template<typename Type >
String Bse::string_from_type (
    Type value )
```

Create a *string* from a templated argument value, such as bool, int, double.

string_from_uint()

```
String Bse::string_from_uint (
    uint64 value )
```

Convert a 64bit unsigned integer into a string.

string_from_unicode() [1/2]

```
std::string Bse::string_from_unicode (
    const uint32_t * codepoints,
    size_t n_codepoints )
```

Convert *codepoints* into an UTF-8 string, using the shortest possible encoding.

string_from_unicode() [2/2]

```
std::string Bse::string_from_unicode (
    const std::vector< uint32_t > & codepoints )
```

Convert *codepoints* into an UTF-8 string, using the shortest possible encoding.

string_has_int()

```
bool Bse::string_has_int (
    const String & string )
```

Checks if a string contains a digit, optionally preceded by whitespaces.

string_hexdump()

```
String Bse::string_hexdump (
    const void * addr,
    size_t length,
    size_t initial_offset )
```

Produce hexdump of a memory region.

Each output line consists of its hexadecimal offset, 16 hexadecimal bytes and the ASCII representation of the same 16 bytes.

string_is_canonified()

```
bool Bse::string_is_canonified (
    const String & string,
    const String & valid_chars )
```

Check if [string_canonify\(\)](#) would modify *string*.

string_is_uuid()

```
bool Bse::string_is_uuid (
    const String & uuid_string )
```

Returns whether *uuid_string* contains a properly formatted UUID string.

string_join()

```
String Bse::string_join (
    const String & junctor,
    const StringVector & strvec )
```

Join a number of strings.

Join a string vector into a single string, using *junctor* inbetween each pair of strings.

string_locale_format()

```
template<class... Args>
String Bse::string_locale_format (
    const char * format,
    const Args &... args )
```

Formatted printing ala **printf()** into a String, using the current locale.

string_locale_vprintf()

```
String Bse::string_locale_vprintf (
    const char * format,
    va_list vargs )
```

Formatted printing like `string_vprintf` using the current locale.

string_lstrip()

```
String Bse::string_lstrip (
    const String & input )
```

Strip whitespaces from the left of a string.

string_match_identifier()

```
bool Bse::string_match_identifier (
    const String & ident1,
    const String & ident2 )
```

Check equality of strings canonicalized to "[0-9a-z_]+".

string_match_identifier_tail()

```
bool Bse::string_match_identifier_tail (
    const String & ident,
    const String & tail )
```

Variant of `string_match_identifier()` that matches *tail* against *ident* at word boundary.

string_multiply()

```
String Bse::string_multiply (
    const String & s,
    uint64 count )
```

Reproduce a string *s* for *count* times.

string_normalize_nfc()

```
String Bse::string_normalize_nfc (
    const String & src )
```

Yield normalized composed UTF-8 string.

string_normalize_nfd()

```
String Bse::string_normalize_nfd (
    const String & src )
```

Yield normalized decomposed UTF-8 string.

string_normalize_nfkc()

```
String Bse::string_normalize_nfkc (
    const String & src )
```

Formatting stripped normalized composed UTF-8 string.

string_normalize_nfkd()

```
String Bse::string_normalize_nfkd (
    const String & src )
```

Formatting stripped normalized decomposed UTF-8 string.

string_option_check()

```
bool Bse::string_option_check (
    const String & option_string,
    const String & option )
```

Check if an option is set/unset in an options list string.

string_option_get()

```
String Bse::string_option_get (
    const String & option_string,
    const String & option )
```

Retrieve the option value from an options list separated by ':' or ';'.

string_options_split()

```
void Bse::string_options_split (
    const String & option_string,
    vector< String > & option_names,
    vector< String > & option_values,
    const String & empty_default )
```

Split an option list string into name/value pairs.

string_replace()

```
String Bse::string_replace (
    const String & input,
    const String & marker,
    const String & replacement,
    size_t maxn )
```

Replace substring *marker* in *input* with *replacement*, at most *maxn* times.

string_rstrip()

```
String Bse::string_rstrip (
    const String & input )
```

Strip whitespaces from the right of a string.

string_set_a2z()

```
String Bse::string_set_a2z ( )
```

Returns a string containing all of a-z.

string_set_A2Z()

```
String Bse::string_set_A2Z ( )
```

Returns a string containing all of A-Z.

string_set_ascii_alnum()

```
String Bse::string_set_ascii_alnum ( )
```

Returns a string containing all of 0-9, A-Z and a-z.

string_split()

```
StringVector Bse::string_split (
    const String & string,
    const String & splitter,
    size_t maxn )
```

Split a string, using *splitter* as delimiter.

Passing "" as *splitter* will split the string at whitespace positions.

string_split_any()

```
StringVector Bse::string_split_any (
    const String & string,
    const String & splitchars,
    size_t maxn )
```

Split a string, using any of the *splitchars* as delimiter.

Passing "" as *splitter* will split the string between all position.

string_startswith()

```
bool Bse::string_startswith (
    const String & string,
    const String & fragment )
```

Returns whether *string* starts with *fragment*.

string_strip()

```
String Bse::string_strip (
    const String & input )
```

Strip whitespaces from the left and right of a string.

string_substitute_char()

```
String Bse::string_substitute_char (
    const String & input,
    const char match,
    const char subst )
```

Replace all occurances of *match* in *input* with *subst*.

string_to_bool()

```
bool Bse::string_to_bool (
    const String & string,
    bool fallback )
```

Interpret a string as boolean value.

Interpret the string as number, "ON"/"OFF" or distinguish "false"/"true" or "yes"/"no" by starting letter. For empty strings, *fallback* is returned.

string_to_cescape()

```
String Bse::string_to_cescape (
    const String & str )
```

Escape text like a C string.

Returns a string that escapes all characters with a backslash '\' that need escaping in C language string syntax.

string_to_cquote()

```
String Bse::string_to_cquote (
    const String & str )
```

Returns a string as C string including double quotes.

string_to_double() [1/2]

```
double Bse::string_to_double (
    const String & string )
```

Parse a double from a string, trying locale specific characters and POSIX/C formatting.

string_to_double() [2/2]

```
double Bse::string_to_double (
    const char * dblstring,
    const char ** endptr )
```

Similar to `string_to_double(const String&)`, but returns the first failing character position in *endptr*.

string_to_double_vector()

```
vector< double > Bse::string_to_double_vector (
    const String & string )
```

Parse a string into a list of doubles, expects ';' as delimiter.

string_to_int()

```
int64 Bse::string_to_int (
    const String & string,
    size_t * consumed,
    uint base )
```

Parse a string into a 64bit integer, optionally specifying the expected number base.

string_to_type()

```
template<typename Type >
Type Bse::string_to_type (
    const String & string )
```

Convert a *string* to template argument type, such as bool, int, double.

string_to_uint()

```
uint64 Bse::string_to_uint (
    const String & string,
    size_t * consumed,
    uint base )
```

Parse a string into a 64bit unsigned integer, optionally specifying the expected number base.

string_tolower()

```
String Bse::string_tolower (
    const String & str )
```

Convert all string characters into Unicode lower case characters.

string_totitle()

```
String Bse::string_totitle (
    const String & str )
```

Convert all string characters into Unicode title characters.

string_toupper()

```
String Bse::string_toupper (
    const String & str )
```

Convert all string characters into Unicode upper case characters.

string_vector_erase_empty()

```
void Bse::string_vector_erase_empty (
    StringVector & svector )
```

Remove empty elements from a string vector.

string_vector_find()

```
String Bse::string_vector_find (
    const StringVector & svector,
    const String & prefix,
    const String & fallback )
```

Search for *prefix* in *svector* and return the matching element.

If multiple matches are possible, the last one is returned.

Returns

fallback if no match was found.

string_vector_find_value()

```
String Bse::string_vector_find_value (
    const StringVector & svector,
    const String & prefix,
    const String & fallback )
```

Search for *prefix* in *svector* and return remainder of the matching string. If multiple matches are possible, the last one is returned.

Returns

fallback if no match was found.

string_vector_lstrip()

```
void Bse::string_vector_lstrip (
    StringVector & svector )
```

Left-strip all elements of a string vector, see [string_lstrip\(\)](#).

string_vector_rstrip()

```
void Bse::string_vector_rstrip (
    StringVector & svector )
```

Right-strip all elements of a string vector, see [string_rstrip\(\)](#).

string_vector_strip()

```
void Bse::string_vector_strip (
    StringVector & svector )
```

Strip all elements of a string vector, see [string_strip\(\)](#).

string_vprintf()

```
String String String Bse::string_vprintf (
    const char * format,
    va_list args )
```

Formatted printing ala **vprintf()** into a String, using the POSIX/C locale.

text_convert()

```
bool Bse::text_convert (
    const String & to_charset,
    String & output_string,
    const String & from_charset,
    const String & input_string,
    const String & fallback_charset,
    const String & output_mark )
```

Convert a string from one encoding to another.

Convert *input_string* from encoding *from_charset* to *to_charset*, returning *output_string*. Interpret unknown characters according to *fallback_charset*. Use *output_mark* in place of unconvertible characters. Returns whether the conversion was successful.

timestamp_benchmark()

```
uint64 Bse::timestamp_benchmark ( )
```

Returns benchmark timestamp in nano-seconds, clock starts around program startup.

timestamp_format()

```
String Bse::timestamp_format (
    uint64 stamp,
    uint maxlength )
```

Convert *stamp* into a string, adding μ second fractions if space permits.

timestamp_realtime()

```
uint64 Bse::timestamp_realtime ( )
```

Return the current time as uint64 in μ seconds.

timestamp_resolution()

```
uint64 Bse::timestamp_resolution ( )
```

Provide resolution of [timestamp_benchmark\(\)](#) in nano-seconds.

timestamp_startup()

```
uint64 Bse::timestamp_startup ( )
```

Provides the [timestamp_realtime\(\)](#) value from program startup.

unicode_is_assigned()

```
constexpr bool Bse::unicode_is_assigned (
    uint32_t u ) [inline]
```

Return whether *u* matches any of the assigned Unicode planes.

unicode_is_character()

```
constexpr bool Bse::unicode_is_character (
    uint32_t u ) [inline]
```

Return whether *u* is not one of the 66 Unicode noncharacters.

unicode_is_control_code()

```
constexpr bool Bse::unicode_is_control_code (
    uint32_t u ) [inline]
```

Return whether *u* is one of the 65 Unicode control codes.

unicode_is_noncharacter()

```
constexpr bool Bse::unicode_is_noncharacter (
    uint32_t u ) [inline]
```

Return whether *u* is one of the 66 Unicode noncharacters.

unicode_is_private()

```
constexpr bool Bse::unicode_is_private (
    uint32_t u ) [inline]
```

Return whether *u* is in one of the 3 private use areas of Unicode.

unicode_is_valid()

```
constexpr bool Bse::unicode_is_valid (
    uint32_t u ) [inline]
```

Return whether *u* is an allowed Unicode codepoint within 0x10FFFF and not part of a UTF-16 surrogate pair.

url_show()

```
bool Bse::url_show (
    const char * url )
```

Find a suitable WWW user agent (taking user configurations into account) and start it to display *url*. Display *url* via a suitable WWW user agent. Several user agents are tried before giving up.

Returns

True if a user agent could be launched successfully.

utf8_to_unicode() [1/2]

```
size_t Bse::utf8_to_unicode (
    const char * str,
    uint32_t * codepoints )
```

Convert valid UTF-8 sequences to Unicode codepoints, invalid sequences are treated as Latin-1 characters. The array *codepoints* must be able to hold at least as many elements as are characters stored in *str*. Returns the number of codepoints stored in *codepoints*.

utf8_to_unicode() [2/2]

```
size_t Bse::utf8_to_unicode (
    const std::string & str,
    std::vector< uint32_t > & codepoints )
```

Convert valid UTF-8 sequences to Unicode codepoints, invalid sequences are treated as Latin-1 characters. Returns the number of codepoints newly stored in *codepoints*.

utf8len() [1/2]

```
size_t Bse::utf8len (
    const char * str )
```

Count valid UTF-8 sequences, invalid sequences are counted as Latin-1 characters.

utf8len() [2/2]

```
size_t Bse::utf8len (
    const std::string & str )
```

Count valid UTF-8 sequences, invalid sequences are counted as Latin-1 characters.

vector_erase_element()

```
template<class V >
bool Bse::vector_erase_element (
    V & v,
    const typename V::value_type & value )
```

Erase element *value* from **std::vector** *v* if it is present.

vector_erase_iface()

```
template<class V , class O >
bool Bse::vector_erase_iface (
    V & v,
    O * value )
```

Erase element *value* from **std::vector** *v* if it matches a vector elements **iface()**.

version()

```
std::string Bse::version ( )
```

Provide a string containing the BSE library build.

warning()

```
template<class ... Args>
void Bse::warning (
    const char * format,
    const Args &... args )
```

Issue a printf-like warning message.

zintern_decompress()

```
uint8 * Bse::zintern_decompress (
    unsigned int decompressed_size,
    const unsigned char * cdata,
    unsigned int cdata_size )
```

Decompress data via zlib.

Parameters

<i>decompressed_size</i>	exact size of the decompressed data to be returned
<i>cdata</i>	compressed data block
<i>cdata_size</i>	exact size of the compressed data block

Returns

decompressed data block or NULL in low memory situations

Decompress the data from *cdata* of length *cdata_size* into a newly allocated block of size *decompressed_size* which is returned. The returned block needs to be released with [zintern_free\(\)](#). This function is intended to

decompress data which has been compressed with the `packres.py` utility, so no errors should occur during decompression. Consequently, if any error occurs during decompression or if the resulting data block is of a size other than `decompressed_size`, the program will abort with an appropriate error message. If not enough memory could be allocated for decompression, NULL is returned.

zintern_free()

```
void Bse::zintern_free (
    uint8 * dc_data )
Free data returned from zintern_decompress().
```

Variable Documentation

cached_hash_secret

```
uint64_t Bse::cached_hash_secret
Use hash_secret() for access.
```

KAMMER_FREQ

```
const SfiReal Bse::KAMMER_FREQ
Kammer note, representing the kammer frequency's MIDI note value for A' or A4.
```

KAMMER_NOTE

```
const SfiInt Bse::KAMMER_NOTE
Value represents unparsable/unknown notes.
```

KAMMER_OCTAVE

```
const SfiInt Bse::KAMMER_OCTAVE
Pitch Standard, see also: https://en.wikipedia.org/wiki/A440\_\(pitch\_standard\)
```

MAX_OCTAVE

```
Const Bse::MAX_OCTAVE
Octave of MIN_NOTE.
```

MIN_FINE_TUNE

```
const SfiInt Bse::MIN_FINE_TUNE
Octave of MAX_NOTE.
```

MIN_OCTAVE

```
Const Bse::MIN_OCTAVE
Octave number for MIDI A'.
```

1.2 Bse::AnsiColors Namespace Reference

The `AnsiColors` namespace contains utility functions for colored terminal output.

Enumerations

- enum `Colors` { , `RESET` }
ANSI color symbols.

Functions

- void `configure` (Colorize colorize)
Override the environment variable `$BSE_COLOR` (which may contain "always", "never" or "auto").
- bool `colorize_tty` (int fd)
Check whether the tty fd should use colorization, checks `BSE_COLOR` if `fd == -1`.
- `std::string` `color` (Colors acolor, Colors c1, Colors c2, Colors c3, Colors c4, Colors c5, Colors c6)
Return ANSI code for the specified color if stdout & stderr should be colorized, see `colorize_tty()`.
- const `char *` `color_code` (Colors acolor)
Return ANSI code for the specified color.

Detailed Description

The `AnsiColors` namespace contains utility functions for colored terminal output.

Enumeration Type Documentation

Colors

enum `Bse::AnsiColors::Colors`
ANSI color symbols.

Enumerator

RESET	Reset combines BOLD_OFF, ITALICS_OFF, UNDERLINE_OFF, INVERSE_OFF, STRIKETHROUGH_OFF.
-------	--

Function Documentation

color()

```
std::string Bse::AnsiColors::color (
    Colors acolor,
    Colors c1,
    Colors c2,
    Colors c3,
    Colors c4,
    Colors c5,
    Colors c6 )
```

Return ANSI code for the specified color if stdout & stderr should be colorized, see `colorize_tty()`.

color_code()

```
const char * Bse::AnsiColors::color_code (
    Colors acolor )
```

Return ANSI code for the specified color.

colorize_tty()

```
bool Bse::AnsiColors::colorize_tty (
    int fd )
```

Check whether the tty *fd* should use colorization, checks BSE_COLOR if *fd* == -1.

configure()

```
void Bse::AnsiColors::configure (
    Colorize colorize )
```

Override the environment variable \$BSE_COLOR (which may contain "always", "never" or "auto").

1.3 Bse::Lib Namespace Reference

Namespace for implementation internals.

Classes

- class [KeccakF1600](#)
The Keccak-f[1600] Permutation, see the Keccak specification Peeters und Assche (2011) .
- class [ScopedLocale](#)
Class to push a specific locale_t for the scope of its lifetime.
- class [ScopedPosixLocale](#)
Class to push the POSIX/C locale_t (UTF-8) for the scope of its lifetime.
- class [StringFormatter](#)
StringFormatter - sprintf() like string formatting for C++.

Detailed Description

Namespace for implementation internals.

1.4 Bse::Path Namespace Reference

The [Path](#) namespace provides functions for file path manipulation and testing.

Functions

- [String dirname](#) (const [String](#) &path)
Retrieve the directory part of the filename @ path.
- [String basename](#) (const [String](#) &path)
Strips all directory components from path and returns the resulting file name.
- [String realpath](#) (const [String](#) &path)
Resolve links and directory references in path and provide a canonicalized absolute pathname.
- [String abspath](#) (const [String](#) &path, const [String](#) &incwd)
- [bool isabs](#) (const [String](#) &path)
Return whether path is an absolute pathname.
- [bool isdirname](#) (const [String](#) &path)
Return whether path is pointing to a directory component.
- [bool mkdirs](#) (const [String](#) &dirpath, [uint](#) mode)
Create the directories in dirpath with mode, check errno on false returns.
- [String user_home](#) (const [String](#) &username)
Get a user's home directory, uses \$HOME if no username is given.
- [String data_home](#) ()
Get the \$XDG_DATA_HOME directory, see: <https://specifications.freedesktop.org/basedir-spec/latest>.

- `String config_home ()`
Get the `$XDG_CONFIG_HOME` directory, see: <https://specifications.freedesktop.org/basedir-spec/latest>.
- `String cache_home ()`
Get the `$XDG_CACHE_HOME` directory, see: <https://specifications.freedesktop.org/basedir-spec/latest>.
- `String runtime_dir ()`
Get the `$XDG_RUNTIME_DIR` directory, see: <https://specifications.freedesktop.org/basedir-spec/latest>.
- `String config_dirs ()`
Get the `$XDG_CONFIG_DIRS` directory list, see: <https://specifications.freedesktop.org/basedir-spec/latest>.
- `String data_dirs ()`
Get the `$XDG_DATA_DIRS` directory list, see: <https://specifications.freedesktop.org/basedir-spec/latest>.
- `String config_names ()`
Get config names as set with `config_names()`, if unset defaults to `program_alias()`.
- `void config_names (const String &names)`
Set a colon separated list of names for this application to find configuration settings and files.
- `String expand_tilde (const String &path)`
Expand a `"~/` or `"~/user/"` path which refers to user home directories.
- `bool check (const String &file, const String &mode)`
- `bool equals (const String &file1, const String &file2)`
- `String cwd ()`
Return the current working directory, including symlinks used in `$PWD` if available.
- `bool searchpath_contains (const String &searchpath, const String &element)`
Check if `searchpath` contains `element`, a trailing slash searches for directories.
- `String searchpath_find (const String &searchpath, const String &file, const String &mode)`
Find the first file in `searchpath` which matches `mode` (see `check()`).
- `StringVector searchpath_list (const String &searchpath, const String &mode)`
Find all `searchpath` entries matching `mode` (see `check()`).
- `String searchpath_multiply (const String &searchpath, const String &postfixes)`
Yield a new `searchpath` by combining each element of `searchpath` with each element of `postfixes`.

Detailed Description

The `Path` namespace provides functions for file path manipulation and testing.

Function Documentation

abspath()

```
String Bse::Path::abspath (
    const String & path,
    const String & incwd )
```

Parameters

<code>path</code>	a filename path
<code>incwd</code>	optional current working directory

Complete `path` to become an absolute file path. If necessary, `incwd` or the real current working directory is prepended.

basename()

```
String Bse::Path::basename (
    const String & path )
```

Strips all directory components from *path* and returns the resulting file name.

cache_home()

`String Bse::Path::cache_home ()`

Get the \$XDG_CACHE_HOME directory, see: <https://specifications.freedesktop.org/basedir-spec/latest>.

check()

```
bool Bse::Path::check (
    const String & file,
    const String & mode )
```

Parameters

<i>file</i>	possibly relative filename
<i>mode</i>	feature string

Returns

true if *file* adheres to *mode*

Perform various checks on *file* and return whether all checks passed. On failure, `errno` is set appropriately, and `FALSE` is returned. Available features to be checked for are:

- e - *file* must exist
- r - *file* must be readable
- w - *file* must be writable
- x - *file* must be executable
- f - *file* must be a regular file
- d - *file* must be a directory
- l - *file* must be a symbolic link
- c - *file* must be a character device
- b - *file* must be a block device
- p - *file* must be a named pipe
- s - *file* must be a socket.

config_dirs()

`String Bse::Path::config_dirs ()`

Get the \$XDG_CONFIG_DIRS directory list, see: <https://specifications.freedesktop.org/basedir-spec/latest>.

config_home()

`String Bse::Path::config_home ()`

Get the \$XDG_CONFIG_HOME directory, see: <https://specifications.freedesktop.org/basedir-spec/latest>.

config_names() [1/2]

`String Bse::Path::config_names ()`

Get config names as set with `config_names()`, if unset defaults to `program_alias()`.

config_names() [2/2]

`void Bse::Path::config_names (`
 `const String & names)`

Set a colon separated list of names for this application to find configuration settings and files.

cwd()

`String Bse::Path::cwd ()`

Return the current working directory, including symlinks used in \$PWD if available.

data_dirs()

`String Bse::Path::data_dirs ()`

Get the \$XDG_DATA_DIRS directory list, see: <https://specifications.freedesktop.org/basedir-spec/latest>.

data_home()

`String Bse::Path::data_home ()`

Get the \$XDG_DATA_HOME directory, see: <https://specifications.freedesktop.org/basedir-spec/latest>.

dirname()

`String Bse::Path::dirname (`
 `const String & path)`

Retrieve the directory part of the filename @ path.

equals()

`bool Bse::Path::equals (`
 `const String & file1,`
 `const String & file2)`

Parameters

<i>file1</i>	possibly relative filename
<i>file2</i>	possibly relative filename

Returns

TRUE if *file1* and *file2* are equal

Check whether *file1* and *file2* are pointing to the same inode in the same file system on the same device.

expand_tilde()

`String Bse::Path::expand_tilde (`
 `const String & path)`

Expand a "~/" or "~user/" *path* which refers to user home directories.

isabs()

```
bool Bse::Path::isabs (
    const String & path )
```

Return whether *path* is an absolute pathname.

isdirname()

```
bool Bse::Path::isdirname (
    const String & path )
```

Return whether *path* is pointing to a directory component.

mkdirs()

```
bool Bse::Path::mkdirs (
    const String & dirpath,
    uint mode )
```

Create the directories in *dirpath* with *mode*, check *errno* on false returns.

realpath()

```
String Bse::Path::realpath (
    const String & path )
```

Resolve links and directory references in *path* and provide a canonicalized absolute pathname.

runtime_dir()

```
String Bse::Path::runtime_dir ( )
```

Get the `$XDG_RUNTIME_DIR` directory, see: <https://specifications.freedesktop.org/basedir-spec/latest>.

searchpath_contains()

```
bool Bse::Path::searchpath_contains (
    const String & searchpath,
    const String & element )
```

Check if *searchpath* contains *element*, a trailing slash searches for directories.

searchpath_find()

```
String Bse::Path::searchpath_find (
    const String & searchpath,
    const String & file,
    const String & mode )
```

Find the first *file* in *searchpath* which matches *mode* (see [check\(\)](#)).

searchpath_list()

```
StringVector Bse::Path::searchpath_list (
    const String & searchpath,
    const String & mode )
```

Find all *searchpath* entries matching *mode* (see [check\(\)](#)).

searchpath_multiply()

```
String Bse::Path::searchpath_multiply (
    const String & searchpath,
    const String & postfixes )
```

Yield a new searchpath by combining each element of *searchpath* with each element of *postfixes*.

user_home()

```
String Bse::Path::user_home (
    const String & username )
```

Get a *user's* home directory, uses \$HOME if no *username* is given.

1.5 Bse::Procedure Namespace Reference

The [Procedure](#) namespace contains procedure/IDL helpers.

Detailed Description

The [Procedure](#) namespace contains procedure/IDL helpers.

1.6 Bse::Re Namespace Reference

Some `std::regex` wrappers to simplify usage and reduce compilation time.

Detailed Description

Some `std::regex` wrappers to simplify usage and reduce compilation time.

1.7 Bse::Test Namespace Reference

The [Test](#) namespace offers utilities for unit tests.

Classes

- class [Timer](#)
Class for profiling benchmark tests.

Functions

- void [init](#) (`int` *argcp, `char` **argv, const [StringVector](#) &args)
Initialize the Bse core for a test program.
- bool [slow](#) ()
Indicates whether slow tests should be run.
- bool [verbose](#) ()
Indicates whether tests should run verbosely.
- `uint64_t` [random_int64](#) ()
Return random int for reproduceble tests.
- `int64_t` [random_irange](#) (`int64_t` begin, `int64_t` end)
Return random int within range for reproduceble tests.
- `double` [random_float](#) ()
Return random double for reproduceble tests.
- `double` [random_frange](#) (`double` begin, `double` end)
Return random double within range for reproduceble tests.

- **int run** (const [StringVector](#) &test_names)
Run named tests.
- **int run** ()
Run all registered tests.
- **String stringify_arg** (const **char** *a, const **char** *str_a)
== Stringify Args ==

Detailed Description

The [Test](#) namespace offers utilities for unit tests.

The [Test](#) namespace is made available by `#include <bse/testing.hh>`

See also `bse/testing.hh`.

Function Documentation

init()

```
void Bse::Test::init (
    int * argcp,
    char ** argv,
    const StringVector & args )
```

Initialize the [Bse](#) core for a test program.

Initializes [Bse](#) to execute unit tests by calling `parse_settings_and_args()` with args "autonomous=1" and "testing=1" and setting the application name. See also `#$BSE_DEBUG`.

random_float()

```
double Bse::Test::random_float ( )
```

Return random double for reproduceable tests.

random_frange()

```
double Bse::Test::random_frange (
    double begin,
    double end )
```

Return random double within range for reproduceable tests.

random_int64()

```
uint64_t Bse::Test::random_int64 ( )
```

Return random int for reproduceable tests.

random_irange()

```
int64_t Bse::Test::random_irange (
    int64_t begin,
    int64_t end )
```

Return random int within range for reproduceable tests.

run() [1/2]

```
int Bse::Test::run (
    const StringVector & test_names )
```

Run named tests.

run() [2/2]

```
int Bse::Test::run (
    void )
```

Run all registered tests.

slow()

```
bool Bse::Test::slow ( )
```

Indicates whether slow tests should be run.

stringify_arg()

```
String Bse::Test::stringify_arg (
    const char * a,
    const char * str_a ) [inline]
== Stringify Args ==
```

verbose()

```
bool Bse::Test::verbose ( )
```

Indicates whether tests should run verbosely.

1.8 Bse::Xms Namespace Reference

Namespace for XML based serialization.

Classes

- struct [DataConverter](#)
Template to specialize XML attribute conversion for various data types.
- class [Reflink](#)
Representation for an object reference between [SerializableInterface](#) objects.
- class [SerializableInterface](#)
Interface for serializable objects with [Reflink](#) support.
- class [SerializationField](#)
Reference to a storage attribute (or child node) for serialization.
- class [SerializationNode](#)
Representation of a storage node for serialization via [save\(\)](#) and [load\(\)](#)

Detailed Description

Namespace for XML based serialization.

1.9 Sfi Module Reference

The [Sfi](#) namespace contains utilities for synthesis.

Detailed Description

The [Sfi](#) namespace contains utilities for synthesis.

Chapter 2

Class Documentation

2.1 `Bse::AlignedArray< T, ALIGNMENT >` Class Template Reference

Class to maintain an array of aligned memory.

```
#include <bcore.hh>
```

Detailed Description

```
template<class T, int ALIGNMENT>
class Bse::AlignedArray< T, ALIGNMENT >
```

Class to maintain an array of aligned memory.

The documentation for this class was generated from the following file:

- bse/bcore.hh

2.2 `Bse::AlignedPOD< SIZE >` Struct Template Reference

Helper to provide memory for placement new `AlignedPOD<SIZE>` is aligned like `max_align_t` or like `malloc()`-ed memory and provides `SIZE` bytes.

```
#include <randomhash.hh>
```

Detailed Description

```
template<size_t SIZE>
struct Bse::AlignedPOD< SIZE >
```

Helper to provide memory for placement new `AlignedPOD<SIZE>` is aligned like `max_align_t` or like `malloc()`-ed memory and provides `SIZE` bytes.

Idiomatic use is:

```
static AlignedPOD<sizeof (std::string)> pod_mem;
std::string *str = new (&pod_mem) std::string();
```

The documentation for this struct was generated from the following file:

- bse/randomhash.hh

2.3 `Bse::AsyncBlockingQueue< Value >` Class Template Reference

Asynchronous queue to push/pop values across thread boundaries.

```
#include <bcore.hh>
```

Detailed Description

```
template<class Value>
class Bse::AsyncBlockingQueue< Value >
```

Asynchronous queue to push/pop values across thread boundaries.

The [AsyncBlockingQueue](#) is a thread-safe asynchronous queue which blocks in pop() until data is provided through push() from any thread.

The documentation for this class was generated from the following file:

- bse/bcore.hh

2.4 Bse::AutoSeeder Class Reference

[AutoSeeder](#) provides non-deterministic seeding entropy.

```
#include <randomhash.hh>
```

Public Member Functions

- [uint64 operator\(\) \(\) const](#)
Generate non-deterministic 64bit random value.
- [template<typename RandomAccessIterator > void generate \(RandomAccessIterator begin, RandomAccessIterator end\)](#)
Fill the range [begin, end) with random unsigned integer values.

Static Public Member Functions

- [static uint64 random \(\)](#)
Generate non-deterministic 64bit random value.

Detailed Description

[AutoSeeder](#) provides non-deterministic seeding entropy.

Member Function Documentation

generate()

```
template<typename RandomAccessIterator >
void Bse::AutoSeeder::generate (
    RandomAccessIterator begin,
    RandomAccessIterator end ) [inline]
```

Fill the range [begin, end) with random unsigned integer values.

operator()()

```
uint64 Bse::AutoSeeder::operator() ( ) const [inline]
```

Generate non-deterministic 64bit random value.

random()

```
static uint64 Bse::AutoSeeder::random ( ) [inline], [static]
```

Generate non-deterministic 64bit random value.

The documentation for this class was generated from the following file:

- bse/randomhash.hh

2.5 Bse::AuxData Struct Reference

[AuxData](#) - record to describe entity attributes with "key = value" strings.

```
import "bseapi.idl";
```

Inherited by [Bse::AuxDataAndIcon](#).

Public Attributes

- [String](#) entity
Entity that has an auxillary data list.
- [StringSeq](#) attributes
List of "key = value" auxillary data strings.

Detailed Description

[AuxData](#) - record to describe entity attributes with "key = value" strings.

Member Data Documentation

attributes

```
StringSeq Bse::AuxData::attributes
```

List of "key = value" auxillary data strings.

entity

```
String Bse::AuxData::entity
```

Entity that has an auxillary data list.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.6 Bse::AuxDataSeq Struct Reference

[AuxDataSeq](#) - a variable length list of [AuxData](#) records.

```
import "bseapi.idl";
```

Detailed Description

[AuxDataSeq](#) - a variable length list of [AuxData](#) records.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.7 Bse::Blob Class Reference

Binary large object storage container.

```
#include <blob.hh>
```

Public Member Functions

- [String name \(\)](#)
Retrieve the Blob's filename or url.
- `const char * data ()`
Retrieve the Blob's data.
- `const uint8 * bytes ()`
Retrieve the Blob's data as uint8 buffer.
- `size_t size ()`
Retrieve the Blob's data size in bytes.
- [String string \(\)](#)
Copy Blob data into a zero terminated string.
- [Blob \(\)](#)
Construct an empty Blob.
- [Blob \(const String &auto_url\)](#)
Construct Blob from url or filename (auto detected).
- `operator bool () const`
Checks if the Blob contains accessible data.

Static Public Member Functions

- `static Blob from_file (const String &filename)`
Create Blob by loading from filename.
- `static Blob from_url (const String &url)`
Create Blob by opening a url.

Detailed Description

Binary large object storage container.

Constructor & Destructor Documentation

Blob() [1/2]

`Bse::Blob::Blob ()` [explicit]
Construct an empty [Blob](#).

Blob() [2/2]

`Bse::Blob::Blob (`
 `const String & auto_url)` [explicit]
Construct [Blob](#) from url or filename (auto detected).

Member Function Documentation

bytes()

`const uint8 * Bse::Blob::bytes ()`
Retrieve the [Blob](#)'s data as uint8 buffer.

data()

```
const char * Bse::Blob::data ( )
```

Retrieve the [Blob](#)'s data.

from_file()

```
Blob Bse::Blob::from_file (
    const String & filename ) [static]
```

Create [Blob](#) by loading from *filename*.

from_url()

```
Blob Bse::Blob::from_url (
    const String & url ) [static]
```

Create [Blob](#) by opening a *url*.

name()

```
String Bse::Blob::name ( )
```

Retrieve the [Blob](#)'s filename or url.

operator bool()

```
Bse::Blob::operator bool ( ) const [explicit]
```

Checks if the [Blob](#) contains accessible data.

size()

```
size_t Bse::Blob::size ( )
```

Retrieve the [Blob](#)'s data size in bytes.

string()

```
String Bse::Blob::string ( )
```

Copy [Blob](#) data into a zero terminated string.

The documentation for this class was generated from the following files:

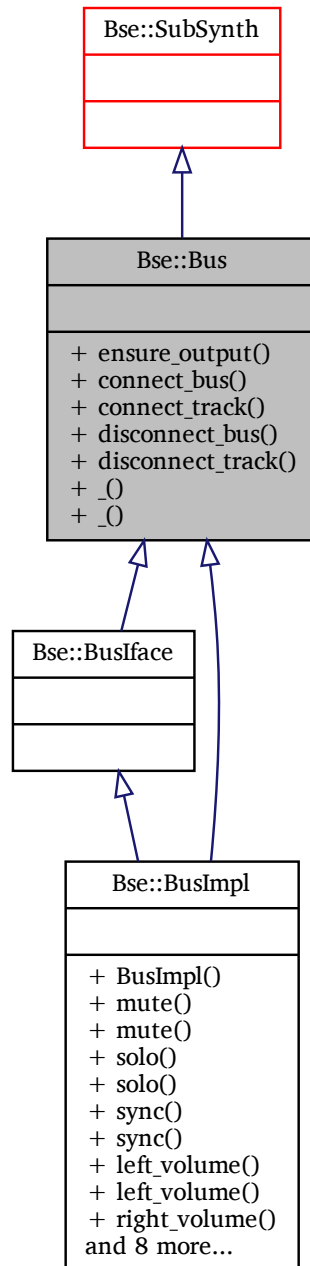
- bse/blob.hh
- bse/blob.cc

2.8 Bse::Bus Interface Reference

Interface for effect stacks and per-track audio signal routing to the master output.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Bus:



Public Member Functions

- Error `ensure_output ()`
Ensure that a bus has an output connection.
- Error `connect_bus (Bus bus)`
Add a bus to the input list of a bus.
- Error `connect_track (Track track)`
Add a track to the input list of a bus.
- Error `disconnect_bus (Bus bus)`

Remove a bus from the input list of a bus.

- Error `disconnect_track` (`Track track`)

Remove a track from the input list of a bus.

Detailed Description

Interface for effect stacks and per-track audio signal routing to the master output.

Member Function Documentation

`connect_bus()`

```
Error Bse::Bus::connect_bus (
    Bus bus )
```

Add a bus to the input list of a bus.

`connect_track()`

```
Error Bse::Bus::connect_track (
    Track track )
```

Add a track to the input list of a bus.

`disconnect_bus()`

```
Error Bse::Bus::disconnect_bus (
    Bus bus )
```

Remove a bus from the input list of a bus.

`disconnect_track()`

```
Error Bse::Bus::disconnect_track (
    Track track )
```

Remove a track from the input list of a bus.

`ensure_output()`

```
Error Bse::Bus::ensure_output ( )
```

Ensure that a bus has an output connection.

The documentation for this interface was generated from the following file:

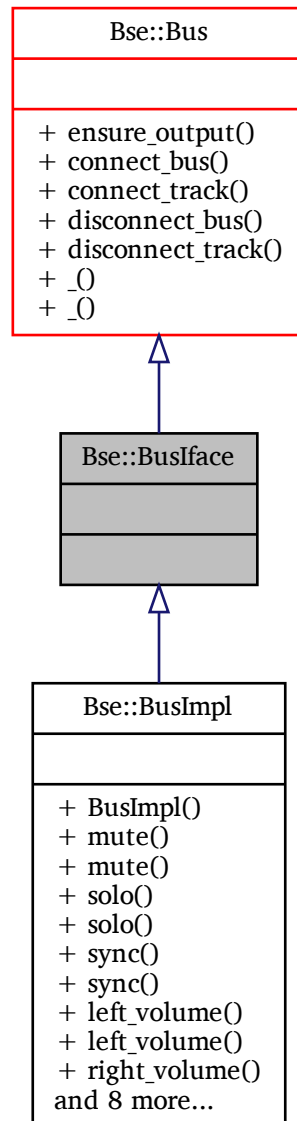
- [bse/bseapi.idl](#)

2.9 Bse::Busiface Class Reference

IDL interface class for `Bse::Bus`.

```
#include <bseapi_interfaces.hh>
```


Inheritance diagram for Bse::Busiface:



Additional Inherited Members

Detailed Description

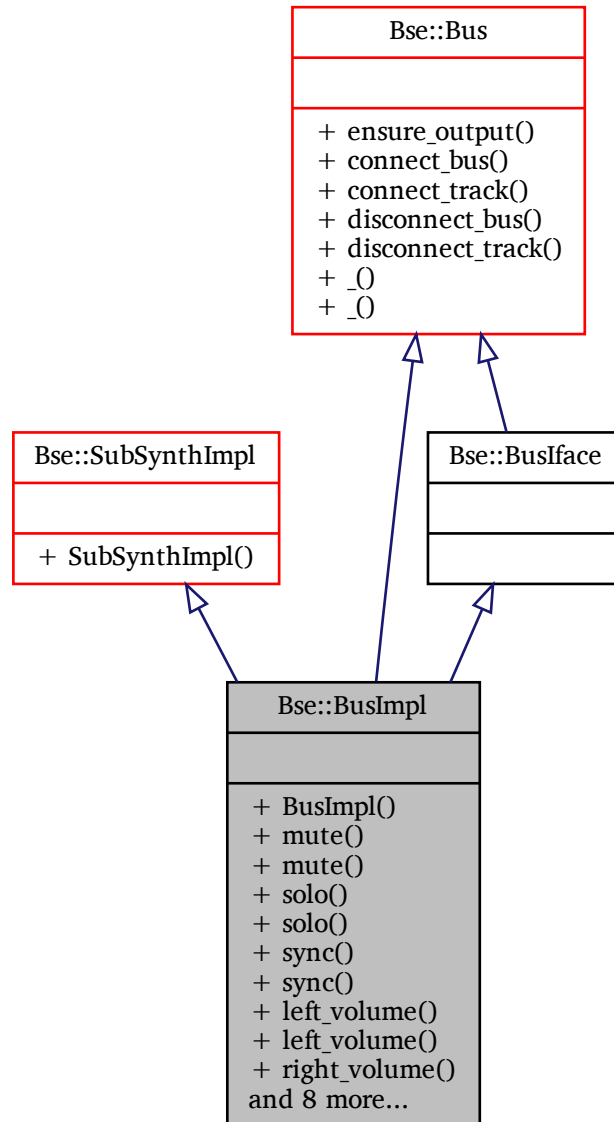
IDL interface class for [Bse::Bus](#).

The documentation for this class was generated from the following file:

- `bse/bseapi_interfaces.hh`

2.10 Bse::BusImpl Class Reference

Inheritance diagram for Bse::BusImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- `bse/bsebus.hh`
- `bse/bsebus.cc`

2.11 Bse::Category Struct Reference

Categories describe useful type entities.

```
import "bseapi.idl";
```

Detailed Description

Categories describe useful type entities.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.12 Bse::CategorySeq Struct Reference

Sequence of [Category](#) records.

```
import "bseapi.idl";
```

Detailed Description

Sequence of [Category](#) records.

The documentation for this struct was generated from the following file:

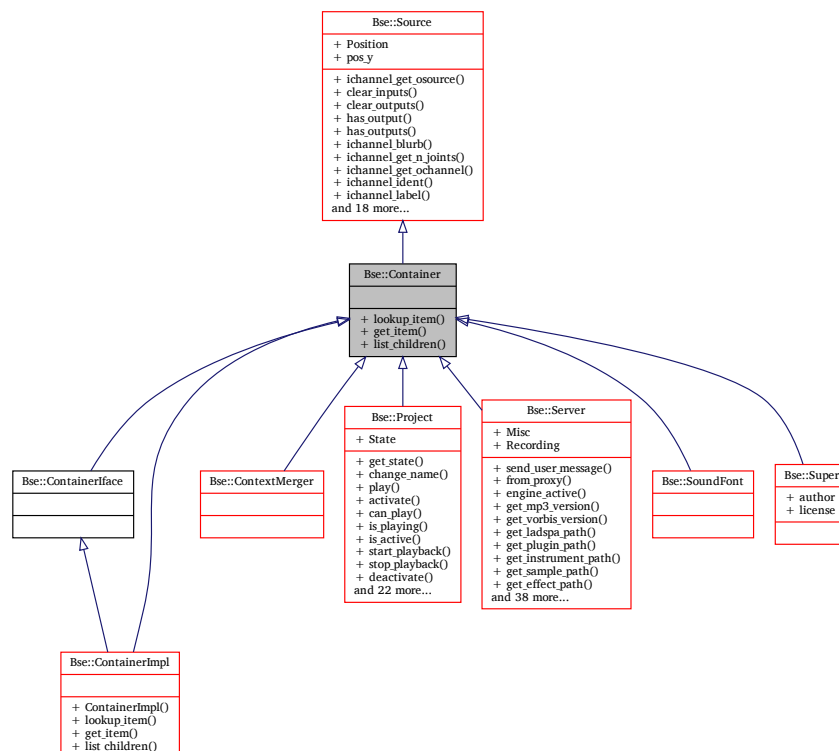
- [bse/bseapi.idl](#)

2.13 Bse::Container Interface Reference

Base interface type for containers of [Item](#) derived types.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Container:



Public Member Functions

- [Item lookup_item](#) (String uname)
Find an immediate child of a container by name (unique per container child).
- [Item get_item](#) (String item_type, int32 seq_id)
Retrieve the immediate child of type item_type by its sequential id (the 'nth' child).
- [ItemSeq list_children](#) ()
List all immediate children of a container.

Detailed Description

Base interface type for containers of [Item](#) derived types.

Events:

- **treechange** - Detail: 'additem' or 'removeitem' - a notification event is sent when an item is added to or removed from the container.

Member Function Documentation

get_item()

```
Item Bse::Container::get_item (
    String item_type,
    int32 seq_id )
```

Retrieve the immediate child of type *item_type* by its sequential id (the 'nth' child).

list_children()

```
ItemSeq Bse::Container::list_children ( )
```

List all immediate children of a container.

lookup_item()

```
Item Bse::Container::lookup_item (
    String uname )
```

Find an immediate child of a container by name (unique per container child).
The documentation for this interface was generated from the following file:

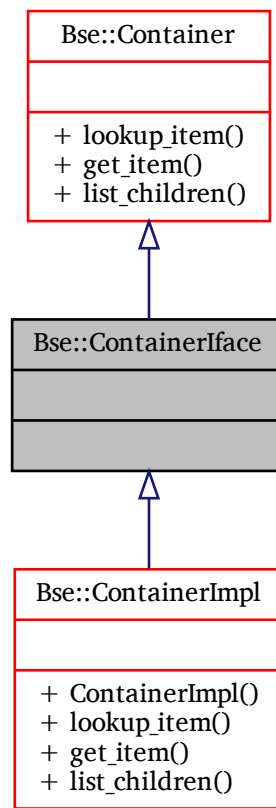
- [bse/bseapi.idl](#)

2.14 Bse::ContainerIface Class Reference

IDL interface class for [Bse::Container](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::ContainerIface:



Additional Inherited Members

Detailed Description

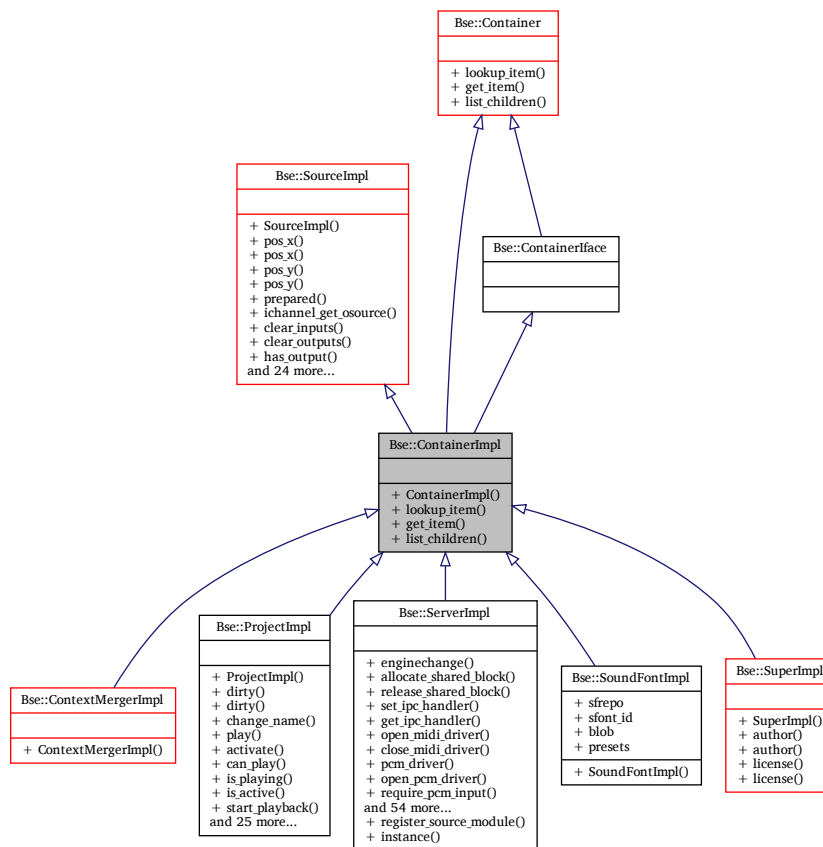
IDL interface class for [Bse::Container](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.15 Bse::ContainerImpl Class Reference

Inheritance diagram for Bse::ContainerImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

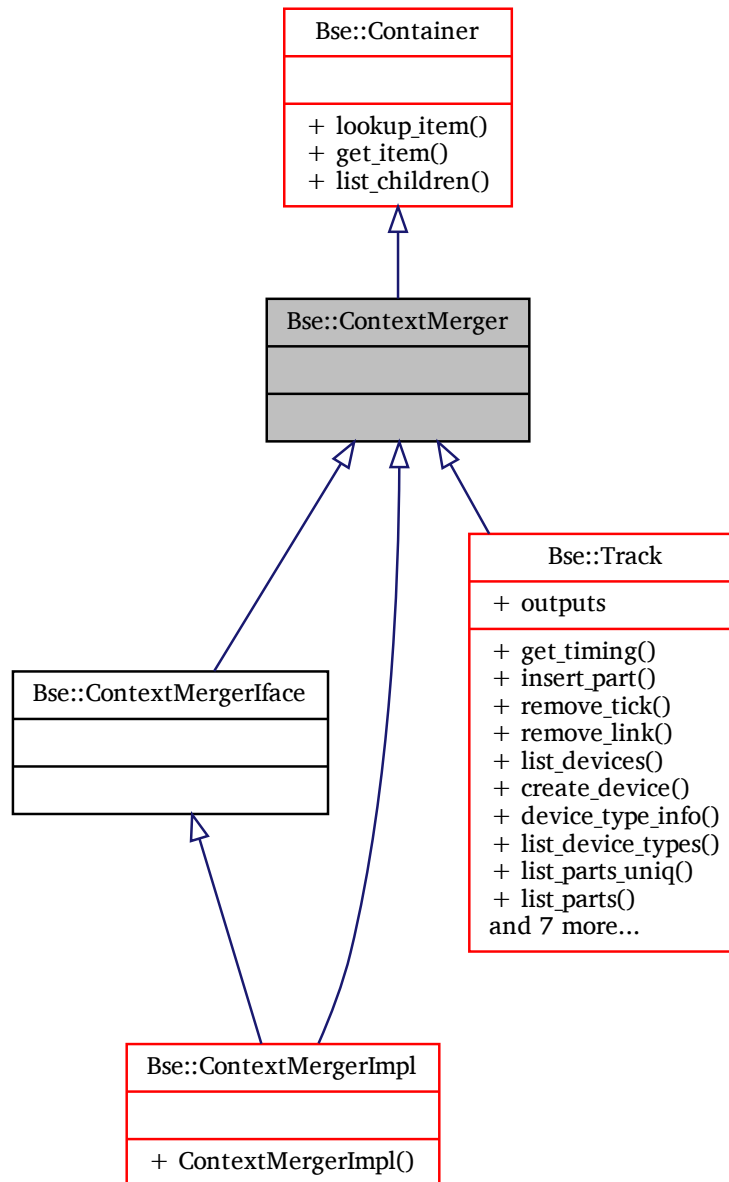
- bse/bsecontainer.hh
- bse/bsecontainer.cc

2.16 Bse::ContextMerger Interface Reference

Source module for merging multiple synthesis contexts, used to implement polyphony.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::ContextMerger:



Additional Inherited Members

Detailed Description

[Source](#) module for merging multiple synthesis contexts, used to implement polyphony. The documentation for this interface was generated from the following file:

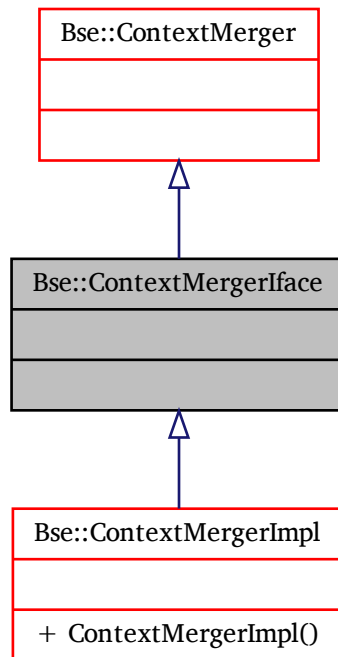
- [bse/bseapi.idl](#)

2.17 Bse::ContextMergerIface Class Reference

IDL interface class for [Bse::ContextMerger](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::ContextMergerIface:



Additional Inherited Members

Detailed Description

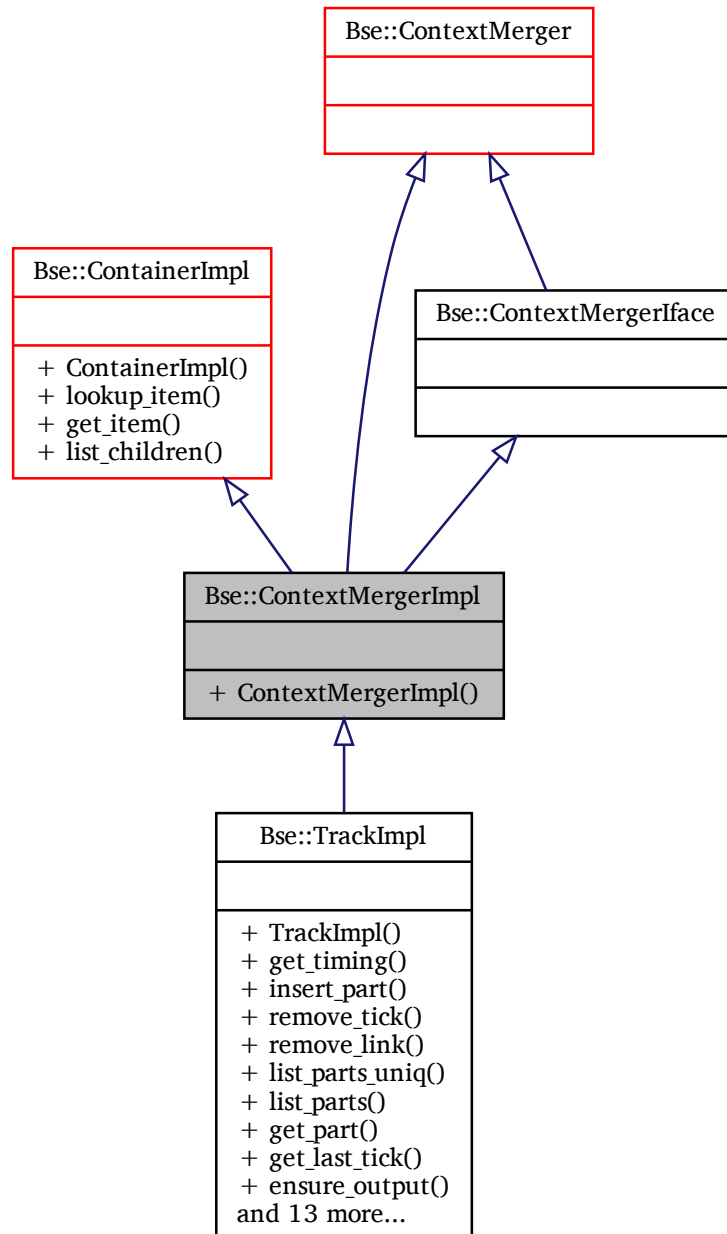
IDL interface class for [Bse::ContextMerger](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.18 Bse::ContextMergerImpl Class Reference

Inheritance diagram for Bse::ContextMergerImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- bse/bsecontextmerger.hh
- bse/bsecontextmerger.cc

2.19 ConvertAny Struct Reference

Convert between Aida::Any and Jsonipc::JsonValue.

```
#include <beast-sound-engine.hh>
```

Inherited by Jsonipc::Convert< Aida::Any >.

Detailed Description

Convert between Aida::Any and Jsonipc::JsonValue.

The documentation for this struct was generated from the following file:

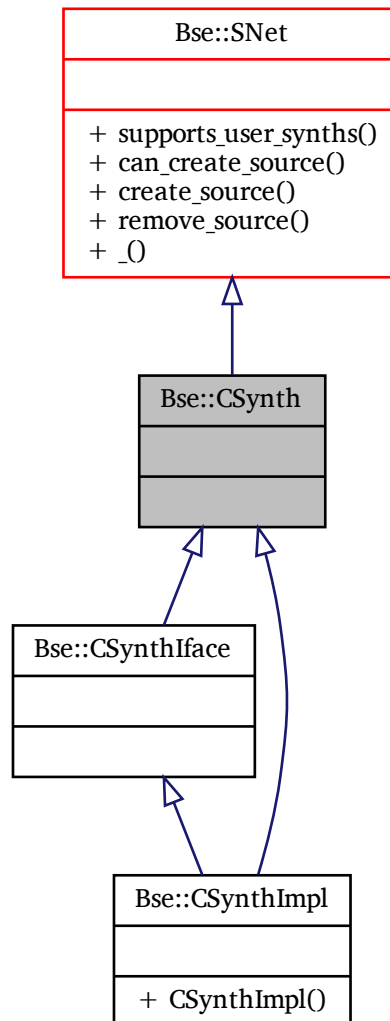
- bse/beast-sound-engine.hh

2.20 Bse::CSynth Interface Reference

Customizable synthesis (filter) network container.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::CSynth:



Additional Inherited Members

Detailed Description

Customizable synthesis (filter) network container.

The documentation for this interface was generated from the following file:

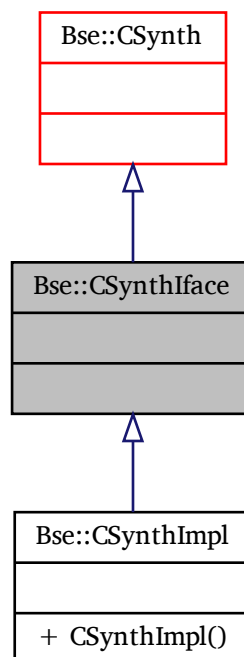
- [bse/bseapi.idl](#)

2.21 Bse::CSynthIface Class Reference

IDL interface class for [Bse::CSynth](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::CSynthIface:



Additional Inherited Members

Detailed Description

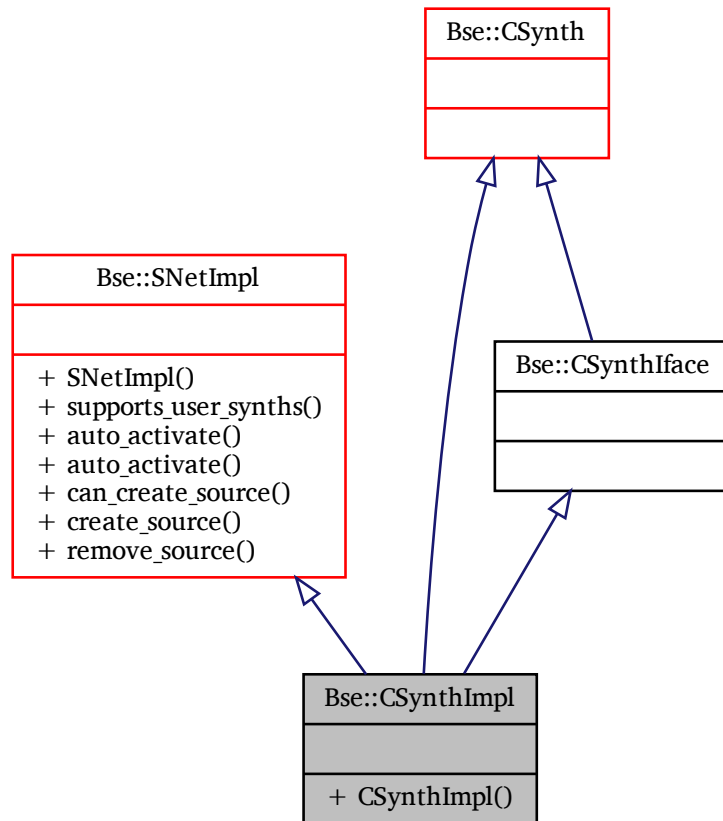
IDL interface class for [Bse::CSynth](#).

The documentation for this class was generated from the following file:

- [bse/bseapi_interfaces.hh](#)

2.22 Bse::CSynthImpl Class Reference

Inheritance diagram for Bse::CSynthImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- bse/bsecsynth.hh
- bse/bsecsynth.cc

2.23 Bse::Xms::DataConverter< T, typename > Struct Template Reference

Template to specialize XML attribute conversion for various data types.

```
#include <serializable.hh>
```

Detailed Description

```
template<typename T, typename = void>
struct Bse::Xms::DataConverter< T, typename >
```

Template to specialize XML attribute conversion for various data types.

The documentation for this struct was generated from the following file:

- bse/serializable.hh

2.24 Bse::DataKey< Type > Class Template Reference

[DataKey](#) objects are used to identify and manage custom data members of [DataListContainer](#) objects.

```
#include <datalist.hh>
```

Public Member Functions

- virtual Type [fallback](#) ()
Return the default value for Type.
- virtual void [destroy](#) (Type data)
Hook invoked when data is deleted.

Detailed Description

```
template<typename Type>
class Bse::DataKey< Type >
```

[DataKey](#) objects are used to identify and manage custom data members of [DataListContainer](#) objects.

Member Function Documentation

destroy()

```
template<typename Type>
virtual void Bse::DataKey< Type >::destroy (
    Type data ) [inline], [virtual]
```

Hook invoked when *data* is deleted.

fallback()

```
template<typename Type>
virtual Type Bse::DataKey< Type >::fallback ( ) [inline], [virtual]
```

Return the default value for Type.

The documentation for this class was generated from the following file:

- bse/datalist.hh

2.25 Bse::DataList Class Reference

Underlying storage implementation for a [DataListContainer](#).

```
#include <datalist.hh>
```

Detailed Description

Underlying storage implementation for a [DataListContainer](#).

The documentation for this class was generated from the following files:

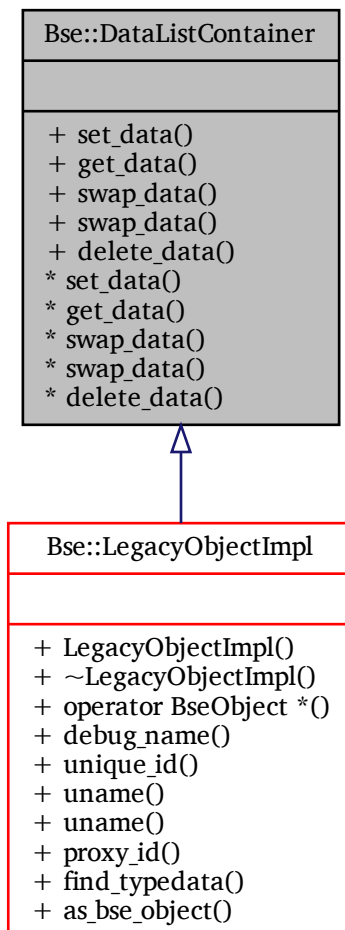
- bse/datalist.hh
- bse/datalist.cc

2.26 Bse::DataListContainer Class Reference

`DataListContainer` - typesafe storage and retrieval of arbitrary members.

```
#include <datalist.hh>
```

Inheritance diagram for `Bse::DataListContainer`:



Public Member Functions

Accessing custom data members

- `template<typename Type >`
`void set_data (DataKey< Type > *key, Type data)`
Assign data to the custom keyed data member, deletes any previously set data.
- `template<typename Type >`
`Type get_data (DataKey< Type > *key) const`
Retrieve contents of the custom keyed data member, returns `DataKey::fallback` if nothing was set.
- `template<typename Type >`
`Type swap_data (DataKey< Type > *key, Type data)`
Swap data with the current contents of the custom keyed data member, returns the current contents.
- `template<typename Type >`
`Type swap_data (DataKey< Type > *key)`
Removes and returns the current contents of the custom keyed data member without deleting it.

- `template<typename Type >`
`void delete_data (DataKey< Type > *key)`
Delete the current contents of the custom keyed data member, invokes `DataKey::destroy`.

Detailed Description

`DataListContainer` - typesafe storage and retrieval of arbitrary members.

By using a `DataKey`, `DataListContainer` objects allow storage and retrieval of custom data members in a typesafe fashion. The custom data members will initially default to `DataKey::fallback` and are deleted by the `DataListContainer` destructor. Example:

Member Function Documentation

`delete_data()`

```
template<typename Type >
void Bse::DataListContainer::delete_data (
    DataKey< Type > * key ) [inline]
```

Delete the current contents of the custom keyed data member, invokes `DataKey::destroy`.

`get_data()`

```
template<typename Type >
Type Bse::DataListContainer::get_data (
    DataKey< Type > * key ) const [inline]
```

Retrieve contents of the custom keyed data member, returns `DataKey::fallback` if nothing was set.

`set_data()`

```
template<typename Type >
void Bse::DataListContainer::set_data (
    DataKey< Type > * key,
    Type data ) [inline]
```

Assign `data` to the custom keyed data member, deletes any previously set data.

`swap_data()` [1/2]

```
template<typename Type >
Type Bse::DataListContainer::swap_data (
    DataKey< Type > * key,
    Type data ) [inline]
```

Swap `data` with the current contents of the custom keyed data member, returns the current contents.

`swap_data()` [2/2]

```
template<typename Type >
Type Bse::DataListContainer::swap_data (
    DataKey< Type > * key ) [inline]
```

Removes and returns the current contents of the custom keyed data member without deleting it.

The documentation for this class was generated from the following file:

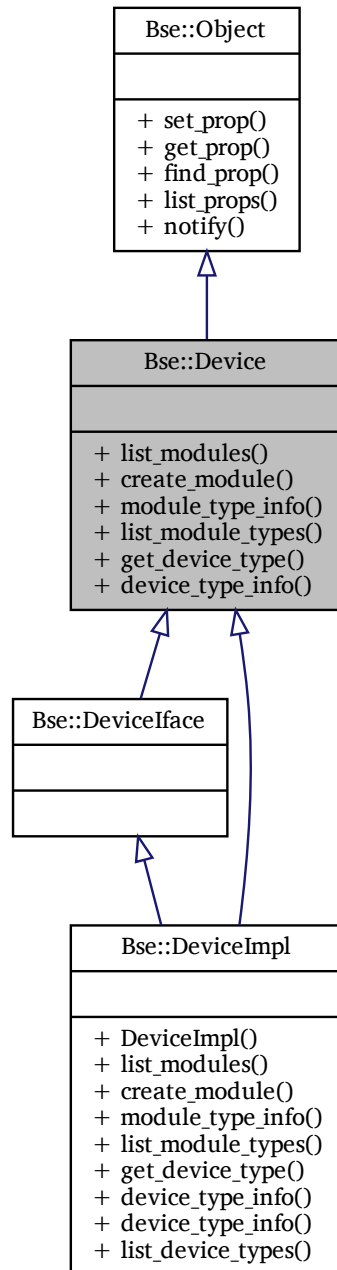
- `bse/datalist.hh`

2.27 Bse::Device Interface Reference

Interface for the encapsulation of audio processors.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Device:



Public Member Functions

- ModuleSeq [list_modules](#) ()
List modules in order of processing.
- Module [create_module](#) (String module_id)

- Create a new module with `module_type`.*

 - `ModuleTypeInfo module_type_info (String module_id)`
Describe `module_type`.
- `StringSeq list_module_types ()`
List known module types.
- `String get_device_type ()`
Retrieve type of device to be created.
- `DeviceTypeInfo device_type_info ()`
Describe this device type.

Detailed Description

Interface for the encapsulation of audio processors.

Member Function Documentation

`create_module()`

```
Module Bse::Device::create_module (
    String module_id )
```

Create a new module with `module_type`.

`device_type_info()`

```
DeviceTypeInfo Bse::Device::device_type_info ( )
```

Describe this device type.

`get_device_type()`

```
String Bse::Device::get_device_type ( )
```

Retrieve type of device to be created.

`list_module_types()`

```
StringSeq Bse::Device::list_module_types ( )
```

List known module types.

`list_modules()`

```
ModuleSeq Bse::Device::list_modules ( )
```

List modules in order of processing.

`module_type_info()`

```
ModuleTypeInfo Bse::Device::module_type_info (
    String module_id )
```

Describe `module_type`.

The documentation for this interface was generated from the following file:

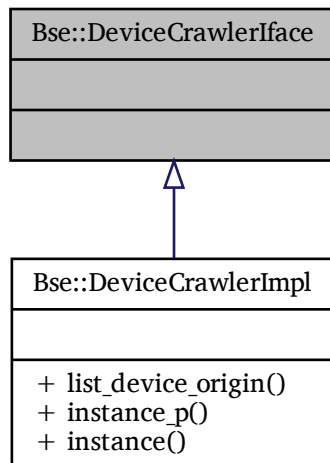
- [bse/bseapi.idl](#)

2.28 Bse::DeviceCrawlerIface Class Reference

IDL interface class for Bse::DeviceCrawler.

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::DeviceCrawlerIface:



Detailed Description

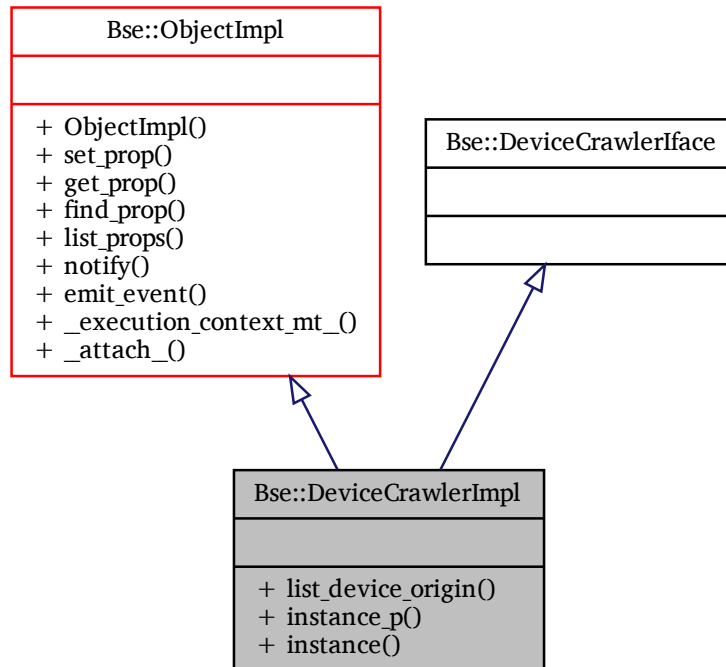
IDL interface class for Bse::DeviceCrawler.

The documentation for this class was generated from the following file:

- `bse/bseapi_interfaces.hh`

2.29 Bse::DeviceCrawlerImpl Class Reference

Inheritance diagram for Bse::DeviceCrawlerImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

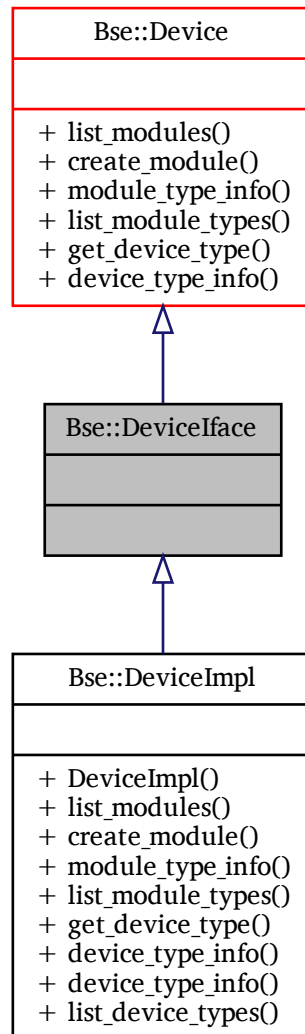
- bse/devicecrawler.hh
- bse/devicecrawler.cc

2.30 Bse::DeviceIface Class Reference

IDL interface class for [Bse::Device](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::DeviceInterface:



Additional Inherited Members

Detailed Description

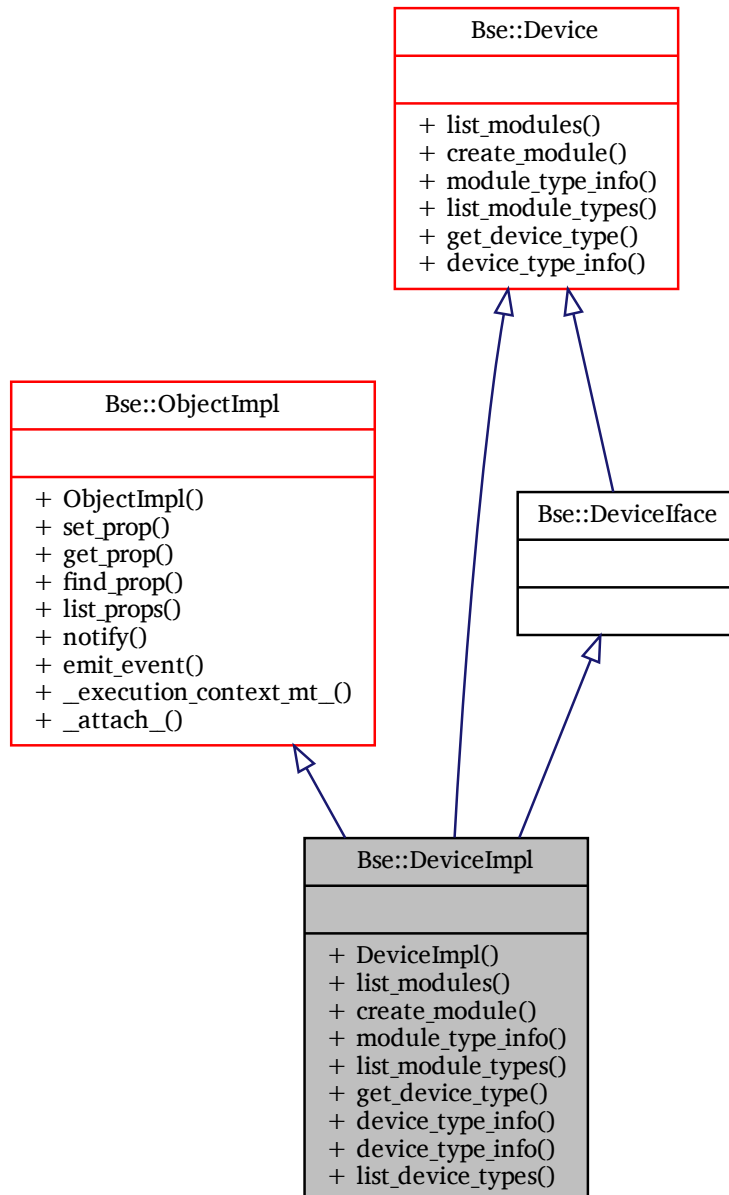
IDL interface class for [Bse::Device](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.31 Bse::DeviceImpl Class Reference

Inheritance diagram for Bse::DeviceImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- bse/device.hh
- bse/device.cc

2.32 Bse::DeviceInfo Struct Reference

Info for device types.
`import "bseapi.idl";`

Detailed Description

Info for device types.
The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.33 Bse::DriverEntry Struct Reference

Driver information for PCM and MIDI handling.
`import "bseapi.idl";`

Detailed Description

Driver information for PCM and MIDI handling.
The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.34 Bse::DriverEntrySeq Struct Reference

[DriverEntry](#) sequence.
`import "bseapi.idl";`

Detailed Description

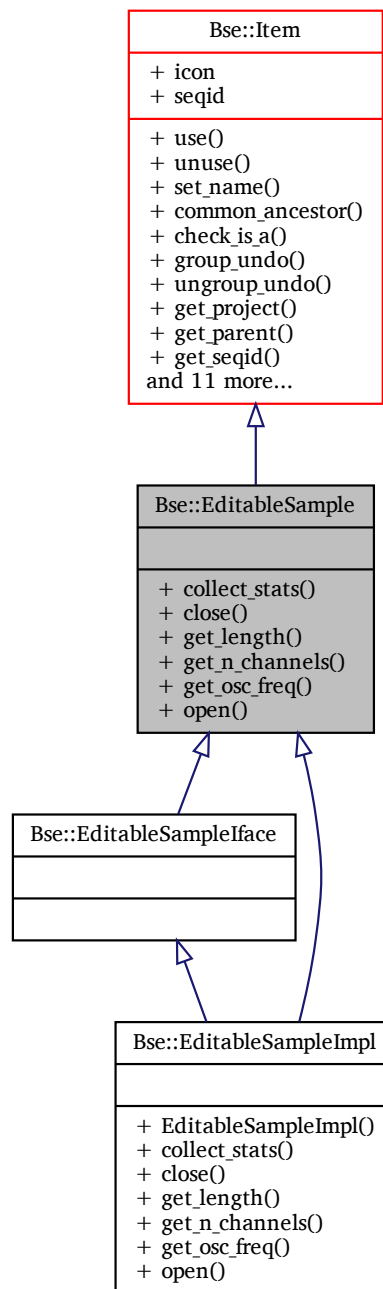
[DriverEntry](#) sequence.
The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.35 Bse::EditableSample Interface Reference

Interface for editable PCM wave samples.
`import "bseapi.idl";`

Inheritance diagram for Bse::EditableSample:



Public Member Functions

- `FloatSeq collect_stats (int64 voffset, float64 offset_scale, int64 block_size, int64 stepping, int64 max_pairs)`
 Collect statistics from sample blocks as (minimum, maximum) pairs.
- `void close ()`
 Close an opened sample.
- `int64 get_length ()`
 Return the number of values in the sample.

- `int64 get_n_channels ()`
Return the number of channels in the sample.
- `float64 get_osc_freq ()`
Return the oscillator frequency for the sample.
- Error `open ()`
Open the sample for reading.

Detailed Description

Interface for editable PCM wave samples.

Member Function Documentation

close()

```
void Bse::EditableSample::close ( )
```

Close an opened sample.

collect_stats()

```
FloatSeq Bse::EditableSample::collect_stats (
    int64 voffset,
    float64 offset_scale,
    int64 block_size,
    int64 stepping,
    int64 max_pairs )
```

Collect statistics from sample blocks as (minimum, maximum) pairs.

Parameters

<i>voffset</i>	Offset of first stat block
<i>offset-scale</i>	Factor to scale voffset increments with
<i>block-size</i>	Block size to compute stat pairs from
<i>stepping</i>	Stepping within a stat block
<i>max-pairs</i>	Maximum number of (min, max) pairs to collect

Returns

Block of samples

get_length()

```
int64 Bse::EditableSample::get_length ( )
```

Return the number of values in the sample.

get_n_channels()

```
int64 Bse::EditableSample::get_n_channels ( )
```

Return the number of channels in the sample.

get_osc_freq()

float64 Bse::EditableSample::get_osc_freq ()
 Return the oscillator frequency for the sample.

open()

Error Bse::EditableSample::open ()

Open the sample for reading.

The documentation for this interface was generated from the following file:

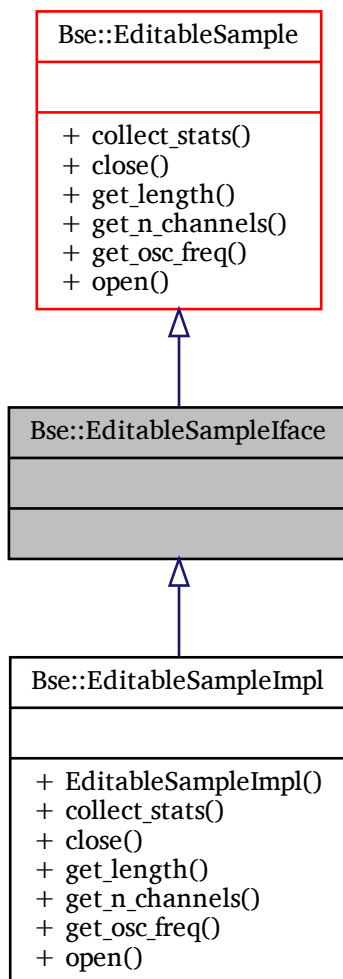
- [bse/bseapi.idl](#)

2.36 Bse::EditableSampleInterface Class Reference

IDL interface class for [Bse::EditableSample](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::EditableSampleInterface:



Additional Inherited Members

Detailed Description

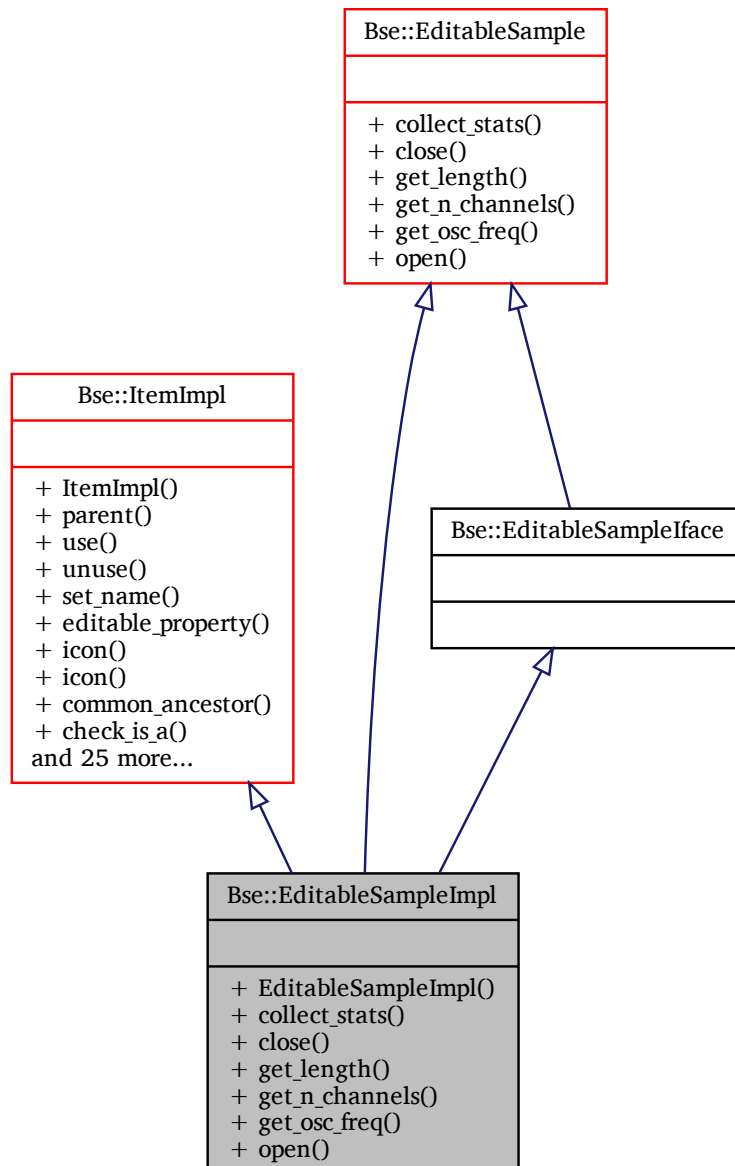
IDL interface class for [Bse::EditableSample](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.37 Bse::EditableSampleImpl Class Reference

Inheritance diagram for Bse::EditableSampleImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- bse/bseeditablesample.hh
- bse/bseeditablesample.cc

2.38 Bse::Flac1Handle Class Reference

[Flac1Handle](#) supports storing flac files as binary appendix to BSE files.

```
#include <bsedatahandle-flac.hh>
```

Public Member Functions

- **int** [read_data](#) (void *buffer, **uint** blength)
Returns -errno || length.
- void [put_wstore](#) (SfiWStore *wstore)
This function deletes the flac1handle object when sfi_wstore_destroy (wstore) is executed.

Static Public Member Functions

- static [Flac1Handle](#) * [create](#) (GslDataHandle *dhandle)
Return a valid [Flac1Handle](#) if dhandle is not flac, and a [Flac1Handle](#) otherwise.

Detailed Description

[Flac1Handle](#) supports storing flac files as binary appendix to BSE files.

Member Function Documentation

create()

```
Flac1Handle * Flac1Handle::create (  
    GslDataHandle * dhandle ) [static]
```

Return a valid [Flac1Handle](#) if *dhandle* is not flac, and a [Flac1Handle](#) otherwise.

put_wstore()

```
void Flac1Handle::put\_wstore (  
    SfiWStore * wstore )
```

This function deletes the flac1handle object when sfi_wstore_destroy (wstore) is executed.

read_data()

```
int Flac1Handle::read\_data (  
    void * buffer,  
    uint blength )
```

Returns -errno || length.

The documentation for this class was generated from the following files:

- bse/bsedatahandle-flac.hh
- bse/bsedatahandle-flac.cc

2.39 Bse::FloatSeq Struct Reference

A list of floating point values.

```
import "bseapi.idl";
```

Detailed Description

A list of floating point values.

The documentation for this struct was generated from the following file:

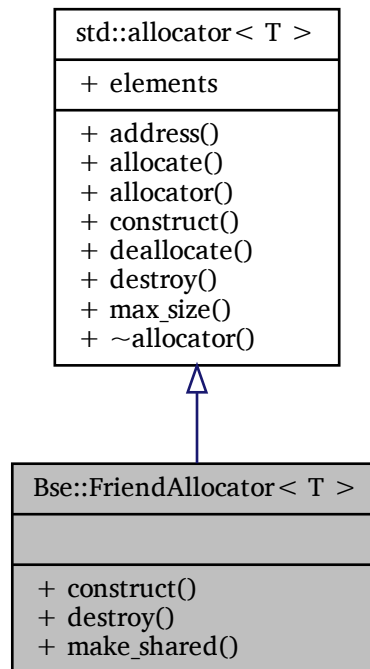
- [bse/bseapi.idl](#)

2.40 Bse::FriendAllocator< T > Class Template Reference

A `std::make_shared<>()` wrapper class to access private ctor & dtor.

```
#include <cxxaux.hh>
```

Inheritance diagram for Bse::FriendAllocator< T >:



Static Public Member Functions

- `template<typename C, typename... Args>`
`static void construct (C *p, Args &&... args)`
Construct type C object, standard allocator requirement.
- `template<typename C >`
`static void destroy (C *p)`
Delete type C object, standard allocator requirement.
- `template<typename ... Args>`
`static std::shared_ptr< T > make_shared (Args &&... args)`
Construct an object of type T that is wrapped into a `std::shared_ptr<T>`.

Additional Inherited Members

Detailed Description

```
template<class T>
class Bse::FriendAllocator< T >
```

A `std::make_shared<>()` wrapper class to access private ctor & dtor.
To call `std::make_shared<T>()` on a class *T*, its constructor and destructor must be public. For classes with private or protected constructor or destructor, this class can be used as follows:

```
class Type {
    Type (ctor_args...);           // Private ctor.
    friend class FriendAllocator<Type>; // Allow access to ctor/dtor of Type.
};
std::shared_ptr<Type> t = FriendAllocator<Type>::make_shared (ctor_args..
.);
```

Member Function Documentation

construct()

```
template<class T >
template<typename C , typename... Args>
static void Bse::FriendAllocator< T >::construct (
    C * p,
    Args &&... args ) [inline], [static]
```

Construct type *C* object, standard allocator requirement.

destroy()

```
template<class T >
template<typename C >
static void Bse::FriendAllocator< T >::destroy (
    C * p ) [inline], [static]
```

Delete type *C* object, standard allocator requirement.

make_shared()

```
template<class T >
template<typename ... Args>
static std::shared_ptr<T> Bse::FriendAllocator< T >::make_shared (
    Args &&... args ) [inline], [static]
```

Construct an object of type *T* that is wrapped into a `std::shared_ptr<T>`.

Parameters

<i>args</i>	The list of arguments to pass into a <i>T()</i> constructor.
-------------	--

Returns

A `std::shared_ptr<T>` owning the newly created object.

The documentation for this class was generated from the following file:

- `bse/cxxaux.hh`

2.41 Bse::Icon Struct Reference

Representation of an icon pixel image.

```
import "bseapi.idl";
```

Detailed Description

Representation of an icon pixel image.

The documentation for this struct was generated from the following file:

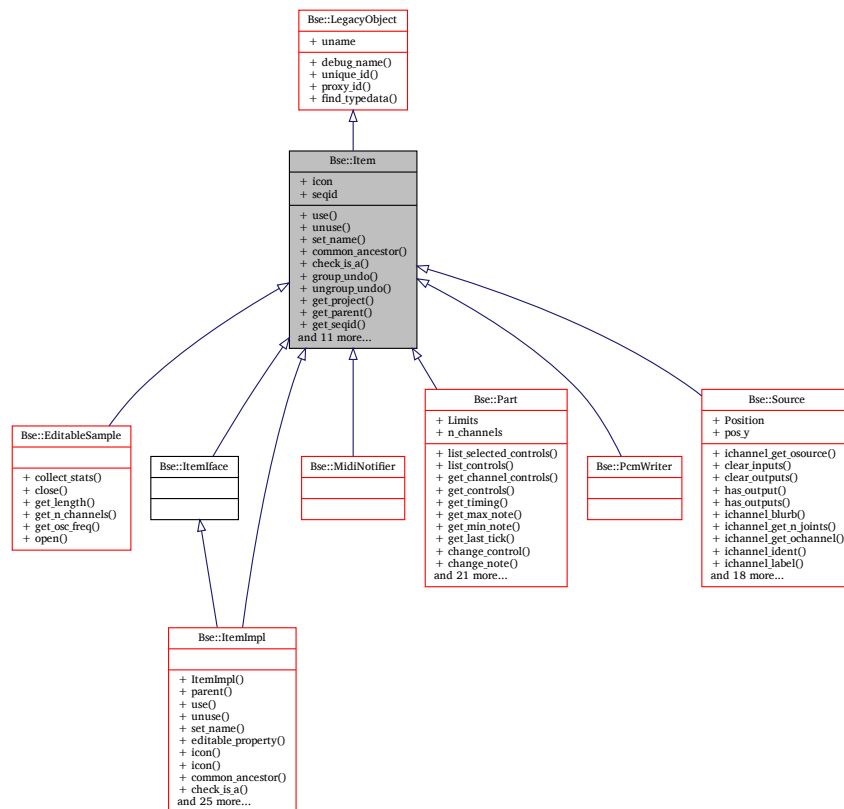
- [bse/bseapi.idl](#)

2.42 Bse::Item Interface Reference

Base interface type for objects that can be added to a container.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Item:



Public Member Functions

- **Item use ()**
Increment use count to keep an item alive.
- **void unuse ()**
Decrement use count for when an item is not needed anymore.
- **void set_name (String name)**
Assign an item's name.
- **Item common_ancestor (Item other)**
Find a common container (parent or grand-parent) of two items if any.
- **bool check_is_a (String type_name)**
Check whether an item has a certain type.

- void `group_undo` (`String` group_name)
Request multiple modifying actions on an item to be grouped together as a single undo operation.
- void `ungroup_undo` ()
Ends the undo grouping opened up by a previous group-undo() call.
- `Project` `get_project` ()
Retrieve an item's project.
- `Item` `get_parent` ()
Retrieve an item's parent.
- `int32` `get_seqid` ()
Retrieve an item's sequential ID. The sequential ID depends on the item's type and its position inbetween siblings of the same type within its immediate container.
- `String` `get_type` ()
Retrieve an item's type name.
- `String` `get_type_authors` ()
Retrieve authors of an item's type implementation.
- `String` `get_type_blurb` ()
Retrieve an item's type description.
- `String` `get_type_license` ()
Retrieve the license for an item's type implementation.
- `String` `get_type_name` ()
Retrieve an item's type name.
- `String` `get_uname_path` ()
Retrieve the project relative uname path for this item.
- `String` `get_name` ()
Retrieve an item's name.
- `String` `get_name_or_type` ()
Retrieve an item's name or type if it has no name.
- bool `internal` ()
Check whether an item is internal, i.e. owned by another non-internal item.
- bool `editable_property` (`String` property)
Test whether a property is editable according to object state and property options.
- `PropertyCandidates` `get_property_candidates` (`String` property_name)
Retrieve tentative values for an item or item sequence property.

Detailed Description

Base interface type for objects that can be added to a container.

Member Function Documentation

`check_is_a()`

```
bool Bse::Item::check_is_a (
    String type_name )
```

Check whether an item has a certain type.

`common_ancestor()`

```
Item Bse::Item::common_ancestor (
    Item other )
```

Find a common container (parent or grand-parent) of two items if any.

editable_property()

```
bool Bse::Item::editable_property (
    String property )
```

Test whether a property is editable according to object state and property options.

get_name()

```
String Bse::Item::get_name ( )
```

Retrieve an item's name.

get_name_or_type()

```
String Bse::Item::get_name_or_type ( )
```

Retrieve an item's name or type if it has no name.

get_parent()

```
Item Bse::Item::get_parent ( )
```

Retrieve an item's parent.

get_project()

```
Project Bse::Item::get_project ( )
```

Retrieve an item's project.

get_property_candidates()

```
PropertyCandidates Bse::Item::get_property_candidates (
    String property_name )
```

Retrieve tentative values for an item or item sequence property.

get_seqid()

```
int32 Bse::Item::get_seqid ( )
```

Retrieve an item's sequential ID. The sequential ID depends on the item's type and its position inbetween siblings of the same type within its immediate container.

get_type()

```
String Bse::Item::get_type ( )
```

Retrieve an item's type name.

get_type_authors()

```
String Bse::Item::get_type_authors ( )
```

Retrieve authors of an item's type implementation.

get_type_blurb()

`String Bse::Item::get_type_blurb ()`
Retrieve an item's type description.

get_type_license()

`String Bse::Item::get_type_license ()`
Retrieve the license for an item's type implementation.

get_type_name()

`String Bse::Item::get_type_name ()`
Retrieve an item's type name.

get_undef_path()

`String Bse::Item::get_undef_path ()`
Retrieve the project relative undef path for this item.

group_undo()

`void Bse::Item::group_undo (`
 `String group_name)`
Request multiple modifying actions on an item to be grouped together as a single undo operation.

internal()

`bool Bse::Item::internal ()`
Check whether an item is internal, i.e. owned by another non-internal item.

set_name()

`void Bse::Item::set_name (`
 `String name)`
Assign an item's name.

ungroup_undo()

`void Bse::Item::ungroup_undo ()`
Ends the undo grouping opened up by a previous group-undo() call.

unuse()

`void Bse::Item::unuse ()`
Decrement use count for when an item is not needed anymore.

use()

`Item Bse::Item::use ()`

Increment use count to keep an item alive.

The documentation for this interface was generated from the following file:

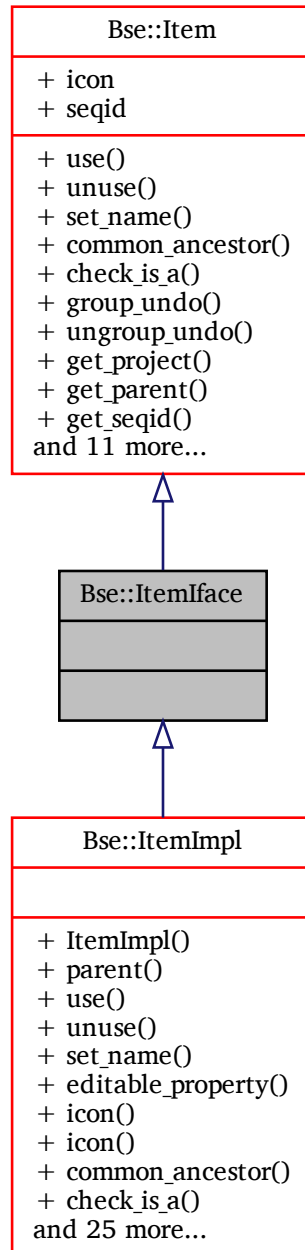
- [bse/bseapi.idl](#)

2.43 Bse::ItemIface Class Reference

IDL interface class for [Bse::Item](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::Itemfcae:



Additional Inherited Members

Detailed Description

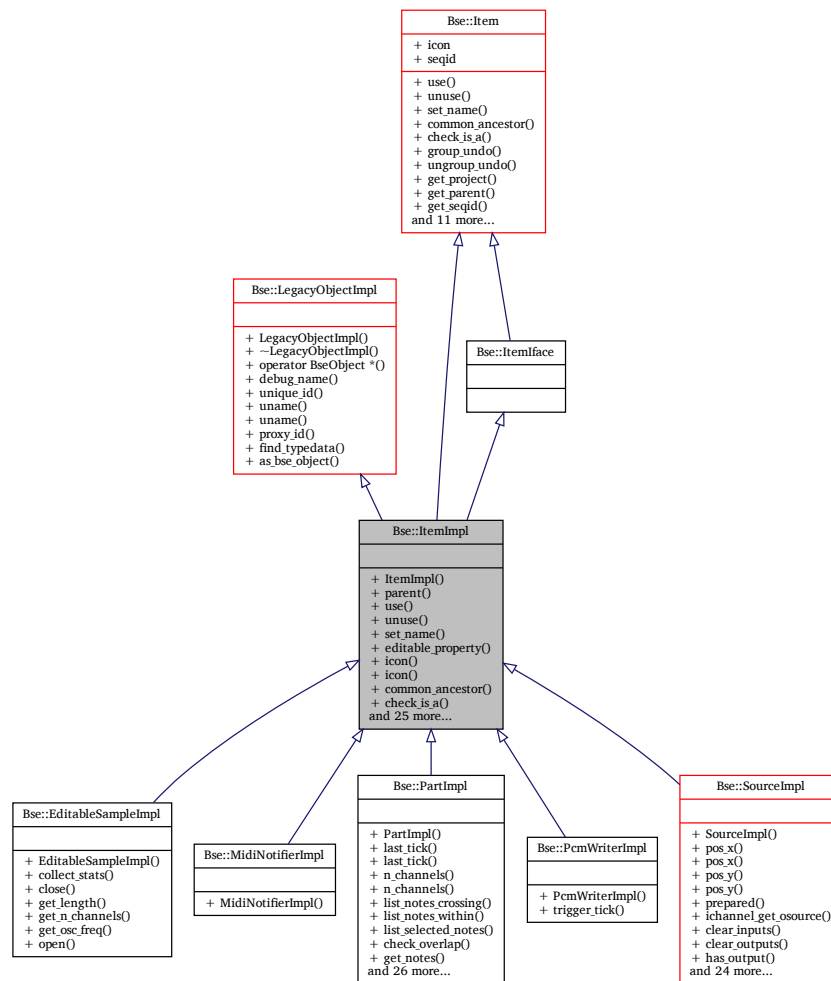
IDL interface class for [Bse::Item](#).

The documentation for this class was generated from the following file:

- [bse/bseapi_interfaces.hh](#)

2.44 Bse::ItemImpl Class Reference

Inheritance diagram for Bse::ItemImpl:



Classes

- class `UndoDescriptor`
UndoDescriptor - type safe object handle to persist undo/redo steps.

Public Member Functions

- void `push_property_undo` (const `String` &property_name)
 Save the value of `property_name` onto the undo stack.
- template<typename ItemT, typename... FuncArgs, typename... CallArgs>
 void `push_undo` (const `String` &blurb, ItemT &self, Error(ItemT::*function)(FuncArgs...), CallArgs... args)
 Push an undo function onto the undo stack, the self argument to function must match this.
- template<typename ItemT, typename R, typename... FuncArgs, typename... CallArgs>
 void `push_undo` (const `String` &blurb, ItemT &self, R(ItemT::*function)(FuncArgs...), CallArgs... args)
 Push an undo function like `push_undo()`, but ignore the return value of function.
- template<typename ItemT, typename ItemTLambda >
 void `push_undo` (const `String` &blurb, ItemT &self, const ItemTLambda &item_lambda)
 Push an undo lambda, using the signature: `Error lambda (TypeDerivedFromItem&, BseUndoStack*)`;

- `template<typename ItemT , typename ItemTLambda >`
`void push_undo_to_redo (const String &blurb, ItemT &self, const ItemTLambda &item_lambda)`
Push an undo step, that when executed, pushes item_lambda to the redo stack.
- `template<class Obj >`
`UndoDescriptor< Obj > undo_descriptor (Obj &item)`
Create an object descriptor that persists undo/redo steps.
- `template<class Obj >`
`Obj & undo_resolve (UndoDescriptor< Obj > udo)`
Resolve an undo descriptor back to an object, see also `undo_descriptor()`.
- `template<class T >`
`bool apply_idl_property (T &lvalue, const T &cvalue, const String &propname)`
Constrain and assign property value if it has changed, emit notification.

Detailed Description

Member Function Documentation

apply_idl_property()

```
template<class T >
bool Bse::ItemImpl::apply_idl_property (
    T & lvalue,
    const T & cvalue,
    const String & propname ) [inline]
```

Constrain and assign property value if it has changed, emit notification.

push_property_undo()

```
void Bse::ItemImpl::push_property_undo (
    const String & property_name )
```

Save the value of *property_name* onto the undo stack.

push_undo() [1/3]

```
template<typename ItemT , typename... FuncArgs, typename... CallArgs>
void Bse::ItemImpl::push_undo (
    const String & blurb,
    ItemT & self,
    Error(ItemT::*)(FuncArgs...) function,
    CallArgs... args ) [inline]
```

Push an undo *function* onto the undo stack, the *self* argument to *function* must match *this*.

push_undo() [2/3]

```
template<typename ItemT , typename R , typename... FuncArgs, typename... CallArgs>
void Bse::ItemImpl::push_undo (
    const String & blurb,
    ItemT & self,
    R(ItemT::*)(FuncArgs...) function,
    CallArgs... args ) [inline]
```

Push an undo *function* like `push_undo()`, but ignore the return value of *function*.

push_undo() [3/3]

```
template<typename ItemT , typename ItemTLambda >
void Bse::ItemImpl::push_undo (
    const String & blurb,
    ItemT & self,
    const ItemTLambda & itemt_lambda ) [inline]
```

Push an undo lambda, using the signature: Error lambda (TypeDerivedFromItem&, BseUndoStack*);.

push_undo_to_redo()

```
template<typename ItemT , typename ItemTLambda >
void Bse::ItemImpl::push_undo_to_redo (
    const String & blurb,
    ItemT & self,
    const ItemTLambda & itemt_lambda ) [inline]
```

Push an undo step, that when executed, pushes *itemt_lambda* to the redo stack.

undo_descriptor()

```
template<class Obj >
UndoDescriptor<Obj> Bse::ItemImpl::undo_descriptor (
    Obj & item ) [inline]
```

Create an object descriptor that persists undo/redo steps.

undo_resolve()

```
template<class Obj >
Obj& Bse::ItemImpl::undo_resolve (
    UndoDescriptor< Obj > udo ) [inline]
```

Resolve an undo descriptor back to an object, see also [undo_descriptor\(\)](#).
The documentation for this class was generated from the following files:

- bse/bseitem.hh
- bse/bseitem.cc

2.45 Bse::ItemSeq Struct Reference

A list of [Item](#) or derived objects.

```
import "bseapi.idl";
```

Detailed Description

A list of [Item](#) or derived objects.

The documentation for this struct was generated from the following file:

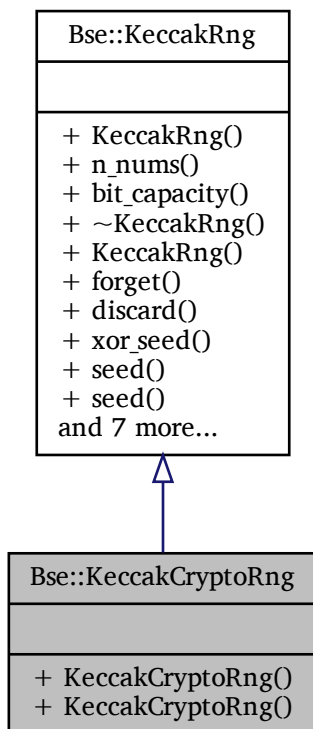
- bse/bseapi.idl

2.46 Bse::KeccakCryptoRng Class Reference

[KeccakCryptoRng](#) - A KeccakF1600 based cryptographic quality pseudo-random number generator.

```
#include <randomhash.hh>
```

Inheritance diagram for Bse::KeccakCryptoRng:



Public Member Functions

- [KeccakCryptoRng \(\)](#)
Initialize and seed the generator from a system specific nondeterministic random source.
- `template<class SeedSeq >`
[KeccakCryptoRng \(SeedSeq &seed_sequence\)](#)
Initialize and seed the generator from seed_sequence.

Additional Inherited Members

Detailed Description

[KeccakCryptoRng](#) - A KeccakF1600 based cryptographic quality pseudo-random number generator. The quality of the generated pseudo random numbers is comaparable to the hash output of [SHAKE128](#).

Constructor & Destructor Documentation

KeccakCryptoRng() [1/2]

`Bse::KeccakCryptoRng::KeccakCryptoRng ()` `[inline]`, `[explicit]`

Initialize and seed the generator from a system specific nondeterministic random source.

KeccakCryptoRng() [2/2]

```
template<class SeedSeq >
Bse::KeccakCryptoRng::KeccakCryptoRng (
    SeedSeq & seed_sequence ) [inline], [explicit]
```

Initialize and seed the generator from *seed_sequence*.

The documentation for this class was generated from the following file:

- bse/randomhash.hh

2.47 Bse::Lib::KeccakF1600 Class Reference

The Keccak-f[1600] Permutation, see the Keccak specification [Peeters und Assche \(2011\)](#).

```
#include <randomhash.hh>
```

Public Member Functions

- [KeccakF1600 \(\)](#)
Zero the state.
- void [reset \(\)](#)
Zero the state.
- void [permute \(uint32_t n_rounds \)](#)
Apply Keccak permutation with n_rounds .
- [uint8_t & byte \(size_t state_index \)](#)

Detailed Description

The Keccak-f[1600] Permutation, see the Keccak specification [Peeters und Assche \(2011\)](#).

Constructor & Destructor Documentation**KeccakF1600()**

```
Bse::Lib::KeccakF1600::KeccakF1600 ( ) [explicit]
```

Zero the state.

Member Function Documentation**byte()**

```
uint8_t& Bse::Lib::KeccakF1600::byte (
    size_t state_index ) [inline]
```

Parameters

<i>state_index</i>	Access byte 0..199 of the state.
--------------------	----------------------------------

permute()

```
void Bse::Lib::KeccakF1600::permute (
    uint32_t n_rounds )
```

Apply Keccak permutation with n_rounds .

The Keccak-f[1600] permutation for up to 254 rounds, see Keccak11 [Peeters und Assche \(2011\)](#) .

reset()

```
void Bse::Lib::KeccakF1600::reset ( )
```

Zero the state.

The documentation for this class was generated from the following files:

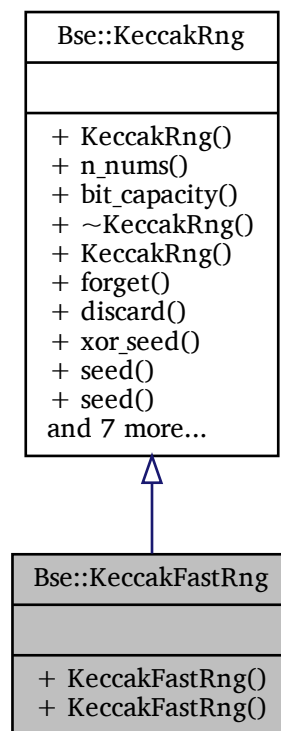
- bse/randomhash.hh
- bse/randomhash.cc

2.48 Bse::KeccakFastRng Class Reference

[KeccakFastRng](#) - A KeccakF1600 based fast pseudo-random number generator.

```
#include <randomhash.hh>
```

Inheritance diagram for Bse::KeccakFastRng:



Public Member Functions

- [KeccakFastRng \(\)](#)
Initialize and seed the generator from a system specific nondeterministic random source.
- `template<class SeedSeq >`
[KeccakFastRng \(SeedSeq &seed_sequence\)](#)
Initialize and seed the generator from seed_sequence.

Additional Inherited Members

Detailed Description

[KeccakFastRng](#) - A KeccakF1600 based fast pseudo-random number generator.

This class tunes the KeccakF1600 algorithm for best performance in pseudo random number generation. Performance is improved while still retaining quality random number generation, according to the findings in section "4.1.1 Statistical tests" from <http://keccak.noekeon.org/Keccak-reference-3.0.pdf>.

Constructor & Destructor Documentation

KeccakFastRng() [1/2]

```
Bse::KeccakFastRng::KeccakFastRng ( ) [inline], [explicit]
```

Initialize and seed the generator from a system specific nondeterministic random source.

KeccakFastRng() [2/2]

```
template<class SeedSeq >
```

```
Bse::KeccakFastRng::KeccakFastRng (
    SeedSeq & seed_sequence ) [inline], [explicit]
```

Initialize and seed the generator from *seed_sequence*.

The documentation for this class was generated from the following file:

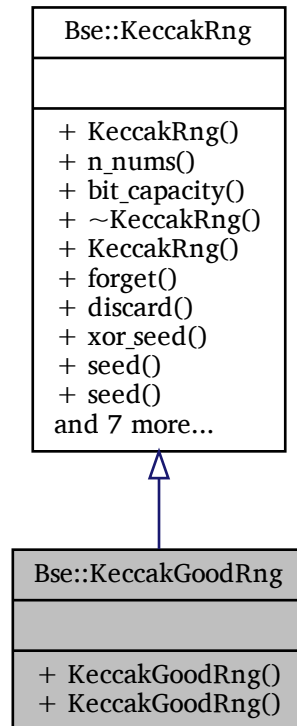
- bse/randomhash.hh

2.49 Bse::KeccakGoodRng Class Reference

[KeccakGoodRng](#) - A KeccakF1600 based good quality pseudo-random number generator.

```
#include <randomhash.hh>
```

Inheritance diagram for Bse::KeccakGoodRng:



Public Member Functions

- [KeccakGoodRng \(\)](#)
Initialize and seed the generator from a system specific nondeterministic random source.
- `template<class SeedSeq >`
[KeccakGoodRng \(SeedSeq &seed_sequence\)](#)
Initialize and seed the generator from seed_sequence.

Additional Inherited Members

Detailed Description

[KeccakGoodRng](#) - A KeccakF1600 based good quality pseudo-random number generator. This class provides very good random numbers, using the KeccakF1600 algorithm without the extra security margins applied for SHA3 hash generation. This improves performance significantly without noticeably trading random number quality. For cryptography grade number generation [KeccakCryptoRng](#) should be used instead.

Constructor & Destructor Documentation

KeccakGoodRng() [1/2]

`Bse::KeccakGoodRng::KeccakGoodRng ()` [`inline`], [`explicit`]

Initialize and seed the generator from a system specific nondeterministic random source.

KeccakGoodRng() [2/2]

```
template<class SeedSeq >
Bse::KeccakGoodRng::KeccakGoodRng (
    SeedSeq & seed_sequence ) [inline], [explicit]
```

Initialize and seed the generator from *seed_sequence*.

The documentation for this class was generated from the following file:

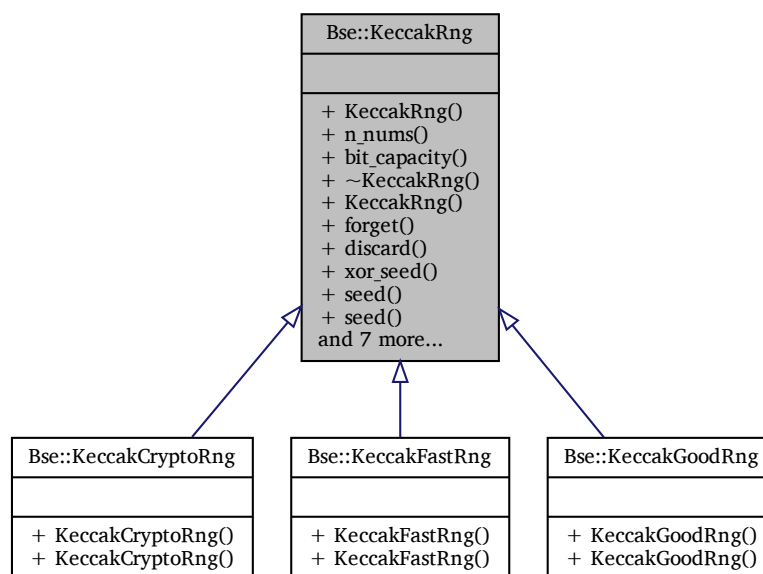
- bse/randomhash.hh

2.50 Bse::KeccakRng Class Reference

KeccakRng - A KeccakF1600 based pseudo-random number generator.

```
#include <randomhash.hh>
```

Inheritance diagram for Bse::KeccakRng:



Public Types

- typedef **uint64_t** *result_type*
Integral type of the KeccakRng generator results.

Public Member Functions

- **size_t** *n_nums* () const
Amount of 64 bit random numbers per generated block.
- **size_t** *bit_capacity* () const
Amount of bits used to store hidden random number generator state.
- **~KeccakRng** ()
The destructor resets the generator state to avoid leaving memory trails.
- **KeccakRng** (**uint16_t** *hidden_state_capacity*, **uint16_t** *n_rounds*)
Create an unseeded Keccak PRNG with specific capacity and number of rounds, for experts only.
- void **forget** ()
Discard 2^{256} bits of the current generator state.

- void `discard` (unsigned `long long` count)
Discard count consecutive random values.
- void `xor_seed` (const `uint64_t` *seeds, `size_t` n_seeds)
Incorporate seed_values into the current generator state.
- void `seed` (`uint64_t` seed_value = 1)
Reinitialize the generator state using a 64 bit seed_value.
- void `seed` (const `uint64_t` *seeds, `size_t` n_seeds)
Reinitialize the generator state using a nuber of 64 bit seeds.
- template<class SeedSeq >
void `seed` (SeedSeq &seed_sequence)
Seed the generator state from a seed_sequence.
- void `auto_seed` ()
Seed the generator from a system specific nondeterministic random source.
- `uint64_t` `random` ()
Generate uniformly distributed 64 bit pseudo random number.
- `result_type` `operator()` ()
Generate uniformly distributed 32 bit pseudo random number.
- template<typename RandomAccessIterator >
void `generate` (RandomAccessIterator begin, RandomAccessIterator end)
Fill the range [begin, end) with random unsigned integer values.
- `result_type` `min` () const
Minimum of the result type, for uint64_t that is 0.
- `result_type` `max` () const
Maximum of the result type, for uint64_t that is 18446744073709551615.

Friends

- bool `operator ==` (const `KeccakRng` &lhs, const `KeccakRng` &rhs)
Compare two generators for state equality.
- bool `operator !=` (const `KeccakRng` &lhs, const `KeccakRng` &rhs)
Compare two generators for state inequality.
- template<typename CharT , typename Traits >
`std::basic_ostream`< CharT, Traits > & `operator<<` (`std::basic_ostream`< CharT, Traits > &os, const `KeccakRng` &self)
Serialize generator state into an OStream.
- template<typename CharT , typename Traits >
`std::basic_istream`< CharT, Traits > & `operator>>` (`std::basic_istream`< CharT, Traits > &is, `KeccakRng` &self)
Deserialize generator state from an IStream.

Detailed Description

`KeccakRng` - A KeccakF1600 based pseudo-random number generator.

The permutation steps are derived from the Keccak specification [Peeters und Assche \(2011\)](#) . For further details about this implementation, see also: <http://testbit.org/keccak> This class is primarily used to implement more fine tuned generators, such as: `KeccakCryptoRng`, `KeccakGoodRng` and `KeccakFastRng`.

Member Typedef Documentation

result_type

```
typedef uint64_t Bse::KeccakRng::result_type
```

Integral type of the `KeccakRng` generator results.

Constructor & Destructor Documentation

~KeccakRng()

```
Bse::KeccakRng::~KeccakRng ( )
```

The destructor resets the generator state to avoid leaving memory trails.

KeccakRng()

```
Bse::KeccakRng::KeccakRng (
    uint16_t hidden_state_capacity,
    uint16_t n_rounds ) [inline], [explicit]
```

Create an unseeded Keccak PRNG with specific capacity and number of rounds, for experts only.

Member Function Documentation

auto_seed()

```
void Bse::KeccakRng::auto_seed ( )
```

Seed the generator from a system specific nondeterministic random source.

bit_capacity()

```
size_t Bse::KeccakRng::bit_capacity ( ) const [inline]
```

Amount of bits used to store hidden random number generator state.

discard()

```
void Bse::KeccakRng::discard (
    unsigned long long count )
```

Discard *count* consecutive random values.

This function is slightly faster than calling `operator()()` exactly *count* times.

forget()

```
void Bse::KeccakRng::forget ( )
```

Discard 2^{256} bits of the current generator state.

This makes it practically infeasible to guess previous generator states or deduce generated values from the past. Use this for forward security [Bellare und Yee \(2001\)](#) of generated security tokens like session keys.

generate()

```
template<typename RandomAccessIterator >
void Bse::KeccakRng::generate (
    RandomAccessIterator begin,
    RandomAccessIterator end ) [inline]
```

Fill the range [begin, end) with random unsigned integer values.

max()

```
result_type Bse::KeccakRng::max ( ) const [inline]
```

Maximum of the result type, for uint64_t that is 18446744073709551615.

min()

`result_type Bse::KeccakRng::min () const [inline]`
 Minimum of the result type, for `uint64_t` that is 0.

n_nums()

`size_t Bse::KeccakRng::n_nums () const [inline]`
 Amount of 64 bit random numbers per generated block.

operator()()

`result_type Bse::KeccakRng::operator() () [inline]`
 Generate uniformly distributed 32 bit pseudo random number.

random()

`uint64_t Bse::KeccakRng::random () [inline]`
 Generate uniformly distributed 64 bit pseudo random number.
 A new block permutation is carried out every `n_nums()` calls, see also `xor_seed()`.

seed() [1/3]

`void Bse::KeccakRng::seed (`
 `uint64_t seed_value = 1) [inline]`
 Reinitialize the generator state using a 64 bit `seed_value`.

seed() [2/3]

`void Bse::KeccakRng::seed (`
 `const uint64_t * seeds,`
 `size_t n_seeds) [inline]`
 Reinitialize the generator state using a number of 64 bit `seeds`.

seed() [3/3]

`template<class SeedSeq >`
`void Bse::KeccakRng::seed (`
 `SeedSeq & seed_sequence) [inline]`
 Seed the generator state from a `seed_sequence`.

xor_seed()

`void Bse::KeccakRng::xor_seed (`
 `const uint64_t * seeds,`
 `size_t n_seeds)`
 Incorporate `seed_values` into the current generator state.
 A block permutation to advance the generator state is carried out per `n_nums()` seed values. After calling this function, generating the next `n_nums()` random values will not need to block for a new permutation.

Friends And Related Function Documentation

operator!=

```
bool operator!= (
    const KeccakRng & lhs,
    const KeccakRng & rhs ) [friend]
```

Compare two generators for state inequality.

operator<<

```
template<typename CharT , typename Traits >
std::basic_ostream<CharT, Traits>& operator<< (
    std::basic_ostream< CharT, Traits > & os,
    const KeccakRng & self ) [friend]
```

Serialize generator state into an OStream.

operator==

```
bool operator== (
    const KeccakRng & lhs,
    const KeccakRng & rhs ) [friend]
```

Compare two generators for state equality.

operator>>

```
template<typename CharT , typename Traits >
std::basic_istream<CharT, Traits>& operator>> (
    std::basic_istream< CharT, Traits > & is,
    KeccakRng & self ) [friend]
```

Deserialize generator state from an IStream.

The documentation for this class was generated from the following files:

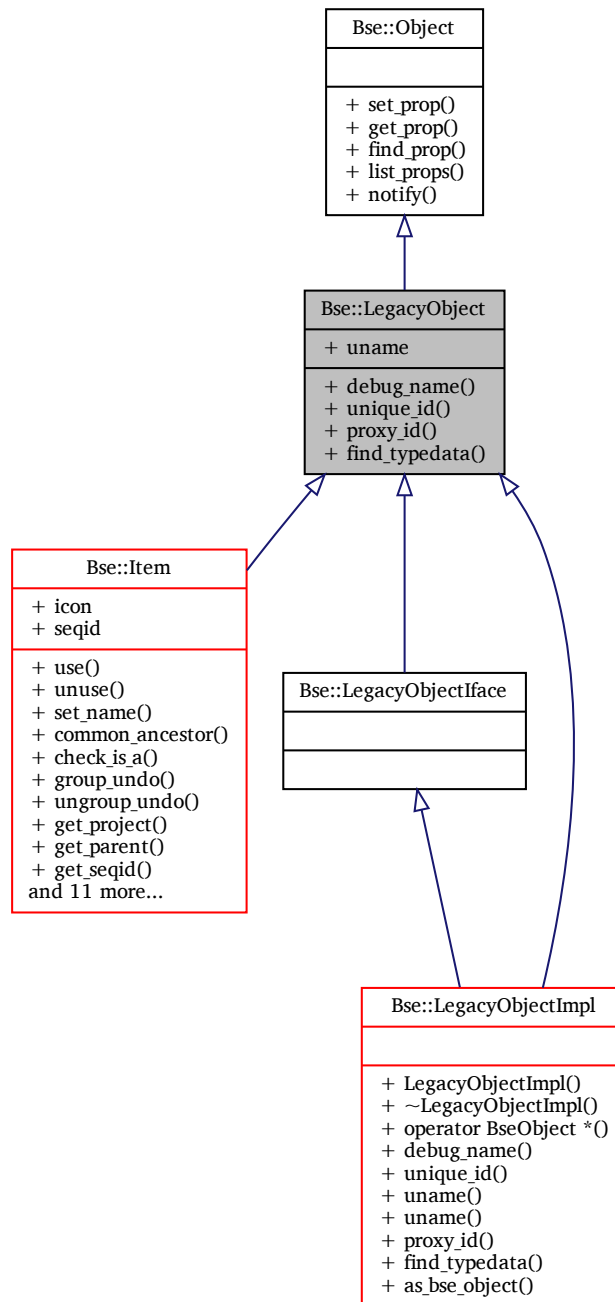
- bse/randomhash.hh
- bse/randomhash.cc

2.51 Bse::LegacyObject Interface Reference

Base type for all legacy objects, derived from struct BseObject.

```
import "bseapi.idl";
```


Inheritance diagram for Bse::LegacyObject:



Public Member Functions

- `String debug_name ()`
Object name useful for debugging output.
- `int32 unique_id ()`
Generate a unique stable ID for this object, similar to a hash of its address.
- `int64 proxy_id ()`
Helper ID for the old SfiGlue interface.
- `StringSeq find_typedata (String type_name)`

Query *key=value* meta info for type *type_name*.

Detailed Description

Base type for all legacy objects, derived from struct `BseObject`.

Events:

- **notify** - A notification event for property changes, the event field *detail* contains the property name.

Member Function Documentation

debug_name()

`String Bse::LegacyObject::debug_name ()`
Object name useful for debugging output.

find_typedata()

`StringSeq Bse::LegacyObject::find_typedata (`
 `String type_name)`
Query *key=value* meta info for type *type_name*.

proxy_id()

`int64 Bse::LegacyObject::proxy_id ()`
Helper ID for the old SfiGlue interface.

unique_id()

`int32 Bse::LegacyObject::unique_id ()`
Generate a unique stable ID for this object, similar to a hash of its address.
The documentation for this interface was generated from the following file:

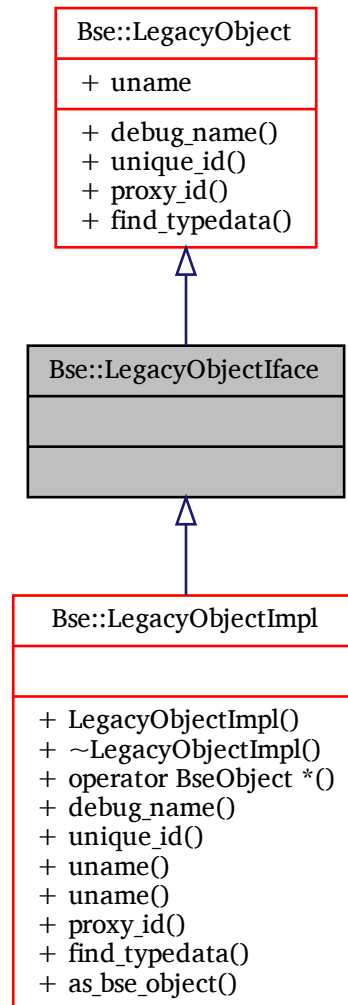
- [bse/bseapi.idl](#)

2.52 Bse::LegacyObjectIface Class Reference

IDL interface class for `Bse::LegacyObject`.

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::LegacyObjectIface:



Additional Inherited Members

Detailed Description

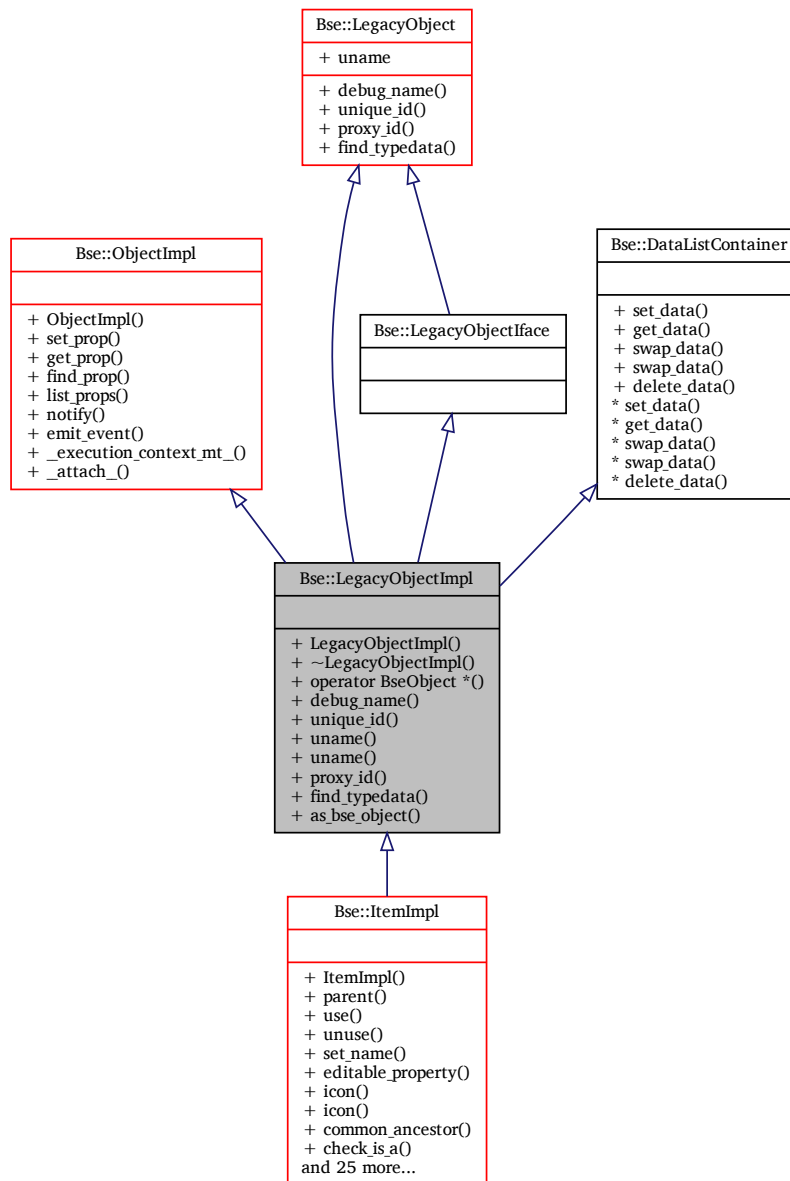
IDL interface class for [Bse::LegacyObject](#).

The documentation for this class was generated from the following file:

- `bse/bseapi_interfaces.hh`

2.53 Bse::LegacyObjectImpl Class Reference

Inheritance diagram for Bse::LegacyObjectImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

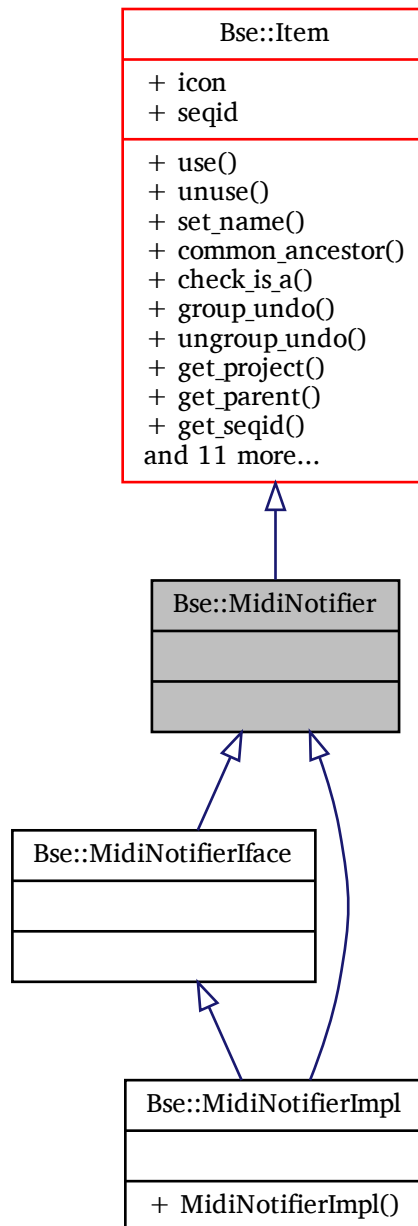
- `bse/bseobject.hh`
- `bse/bseobject.cc`

2.54 Bse::MidiNotifier Interface Reference

Interface for MIDI event notification.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::MidiNotifier:



Additional Inherited Members

Detailed Description

Interface for MIDI event notification.

The documentation for this interface was generated from the following file:

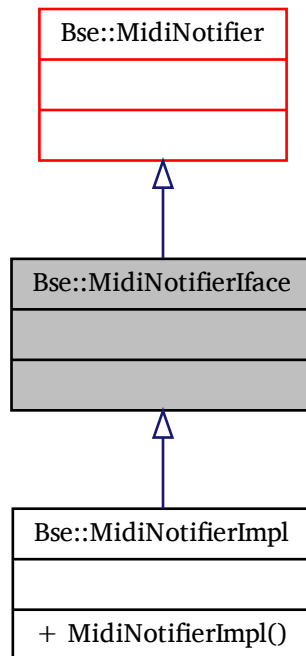
- [bse/bseapi.idl](#)

2.55 Bse::MidiNotifierIface Class Reference

IDL interface class for [Bse::MidiNotifier](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::MidiNotifierIface:



Additional Inherited Members

Detailed Description

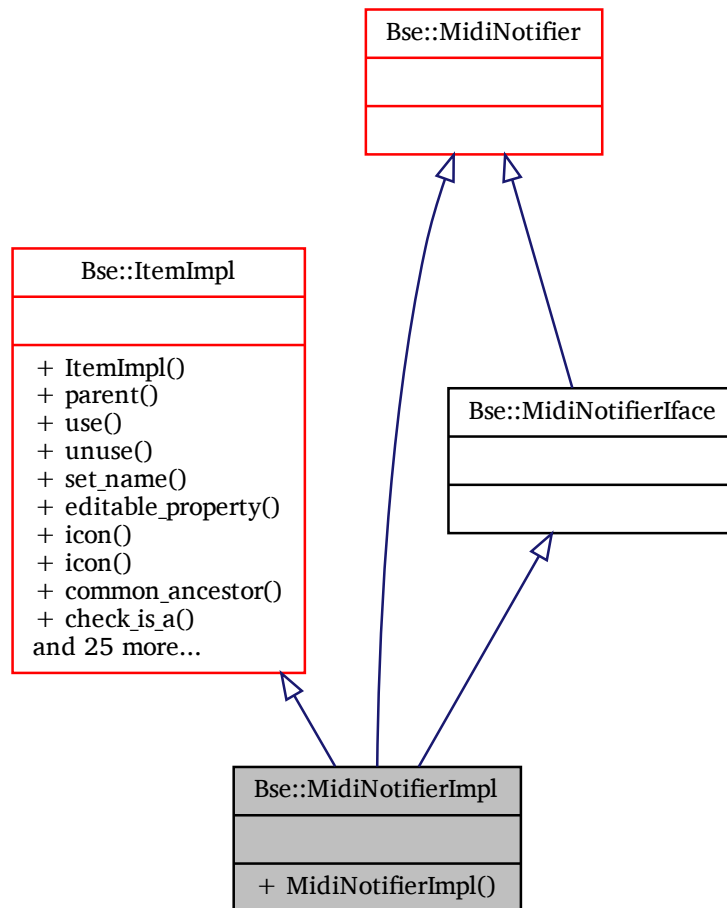
IDL interface class for [Bse::MidiNotifier](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.56 Bse::MidiNotifierImpl Class Reference

Inheritance diagram for Bse::MidiNotifierImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

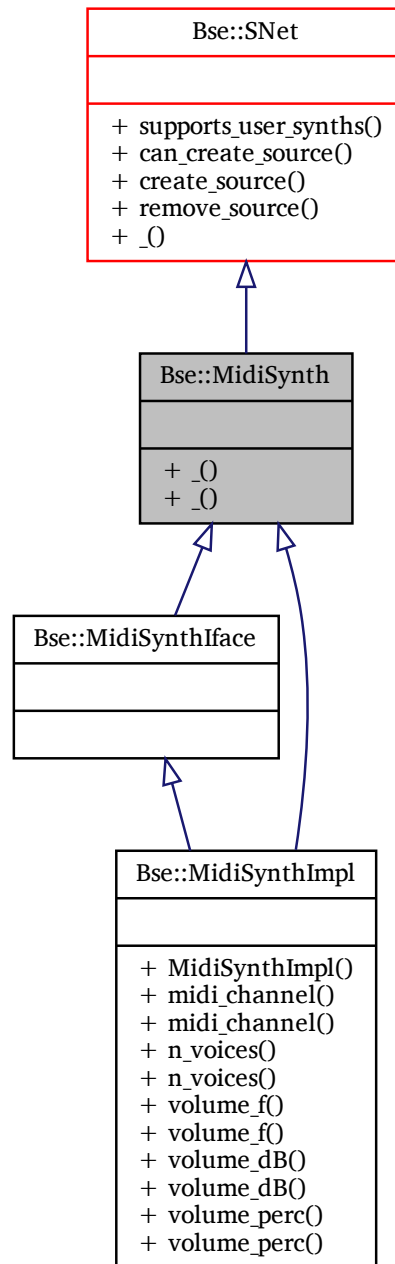
- bse/bsemidinotifier.hh
- bse/bsemidinotifier.cc

2.57 Bse::MidiSynth Interface Reference

Interface for MIDI synthesis networks.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::MidiSynth:



Additional Inherited Members

Detailed Description

Interface for MIDI synthesis networks.

The documentation for this interface was generated from the following file:

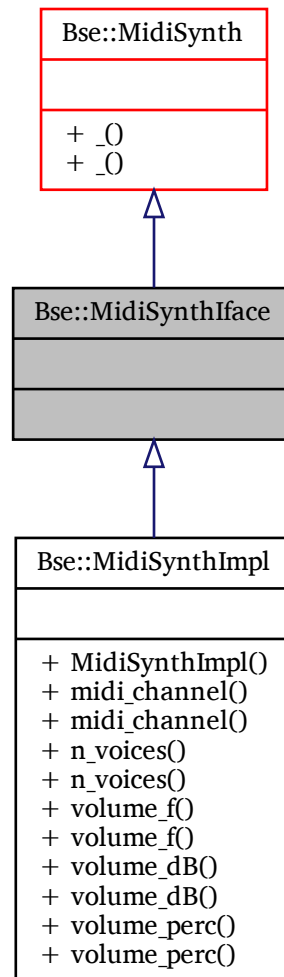
- [bse/bseapi.idl](#)

2.58 Bse::MidiSynthIface Class Reference

IDL interface class for [Bse::MidiSynth](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::MidiSynthIface:



Additional Inherited Members

Detailed Description

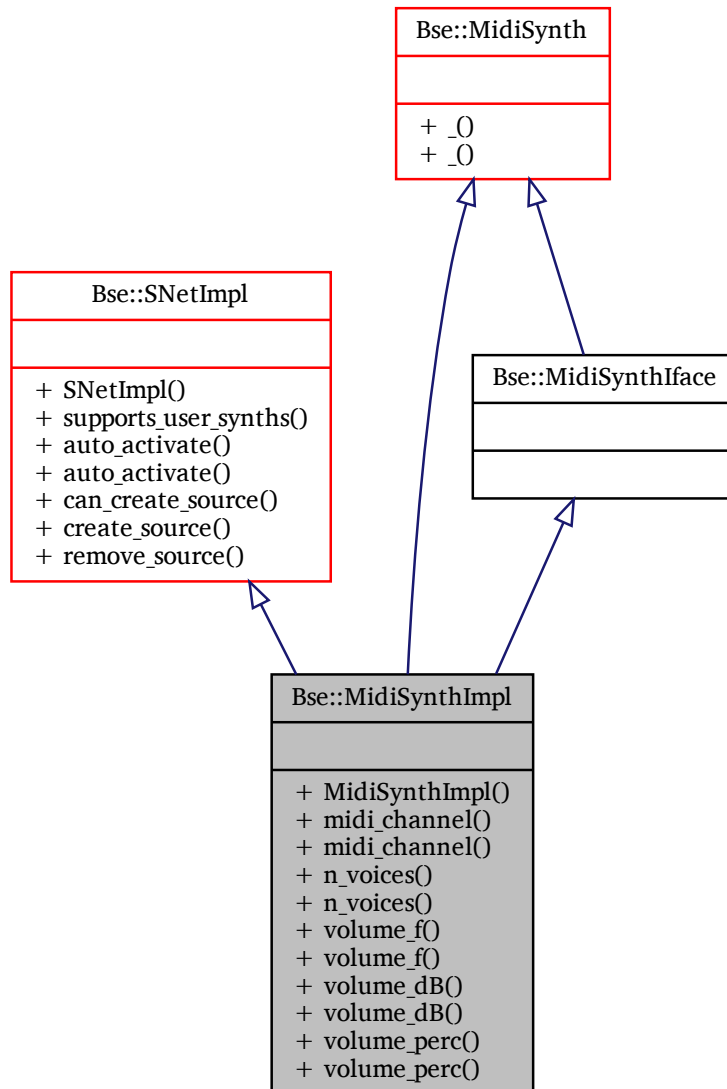
IDL interface class for [Bse::MidiSynth](#).

The documentation for this class was generated from the following file:

- `bse/bseapi_interfaces.hh`

2.59 Bse::MidiSynthImpl Class Reference

Inheritance diagram for Bse::MidiSynthImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

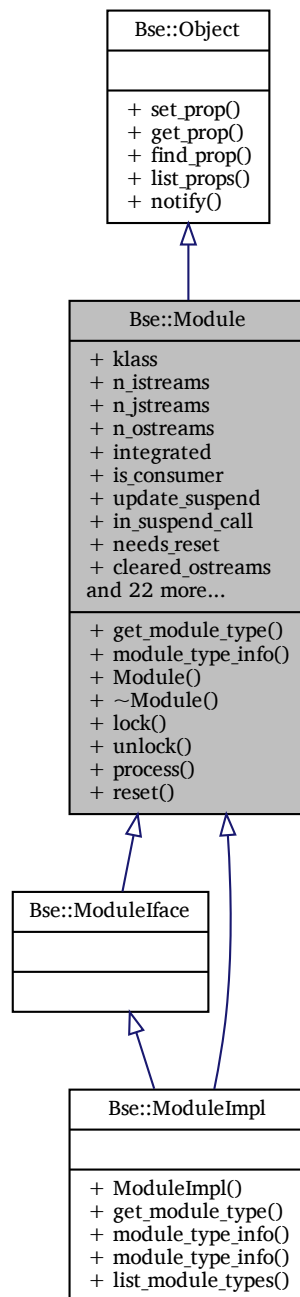
- `bse/bsemidisynth.hh`
- `bse/bsemidisynth.cc`

2.60 Bse::Module Class Reference

Interface for the encapsulation of audio processors.

```
import "bseengine/node.hh";
```

Inheritance diagram for Bse::Module:



Public Member Functions

- [String get_module_type \(\)](#)
Retrieve type of module to be created.
- [ModuleTypeInfo module_type_info \(\)](#)
Describe this module type.

Detailed Description

Interface for the encapsulation of audio processors.
DSP Engine [Module](#).

Member Function Documentation

get_module_type()

`String Bse::Module::get_module_type ()`
Retrieve type of module to be created.

module_type_info()

`ModuleTypeInfo Bse::Module::module_type_info ()`
Describe this module type.

The documentation for this class was generated from the following files:

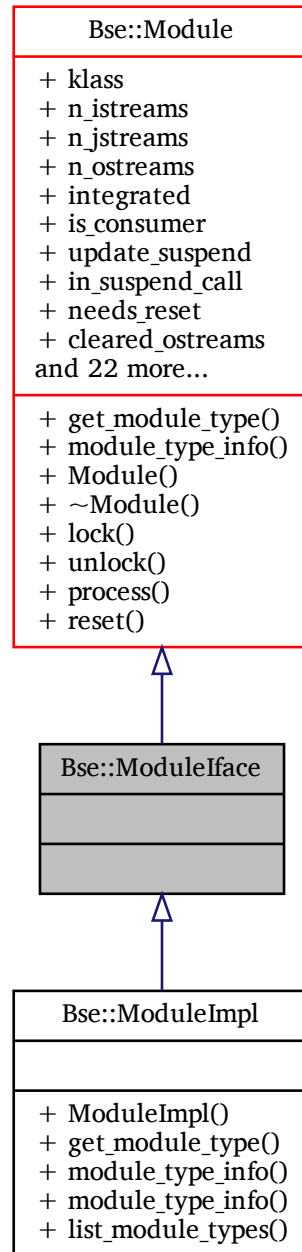
- [bse/bseapi.idl](#)
- [bse/bseenginemode.hh](#)
- [bse/bseengine.cc](#)

2.61 Bse::ModuleIface Class Reference

IDL interface class for [Bse::Module](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::ModuleIface:



Additional Inherited Members

Detailed Description

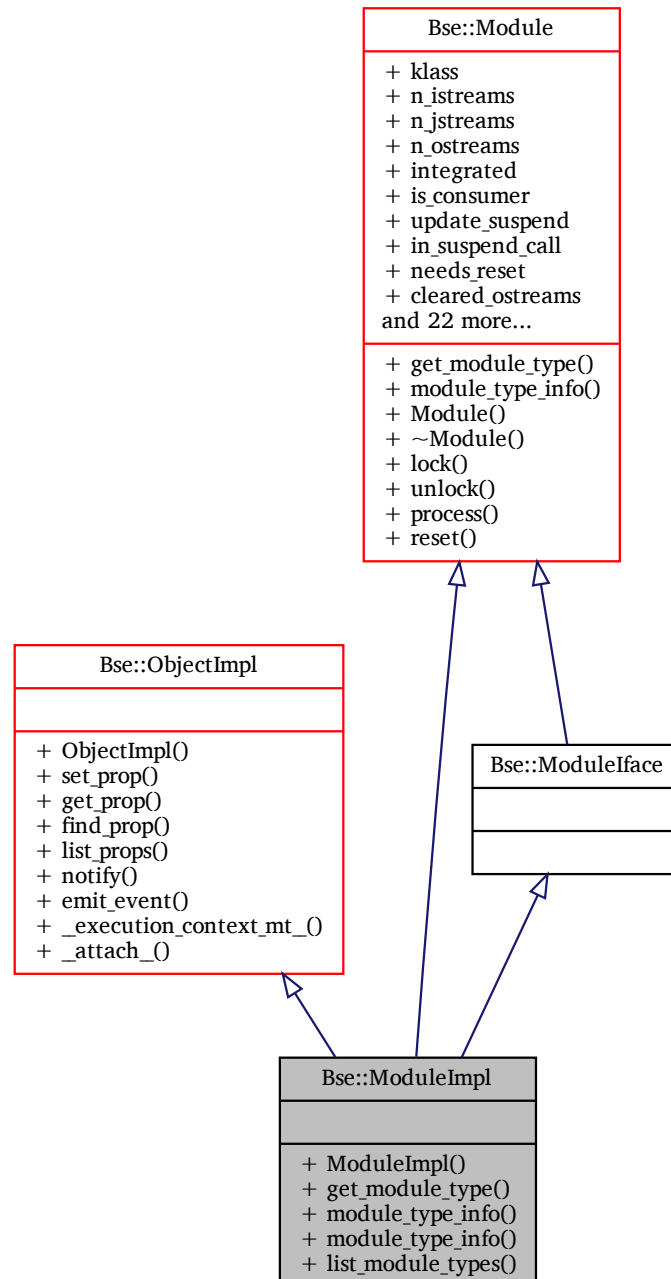
IDL interface class for [Bse::Module](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.62 Bse::ModuleImpl Class Reference

Inheritance diagram for Bse::ModuleImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- bse/module.hh
- bse/module.cc

2.63 Bse::ModuleTypeInfo Struct Reference

Info for module types.
import "bseapi.idl";

Detailed Description

Info for module types.
The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.64 Bse::NoteDescription Struct Reference

A note description provides all needed details about a specific note. ”.
import "bseapi.idl";

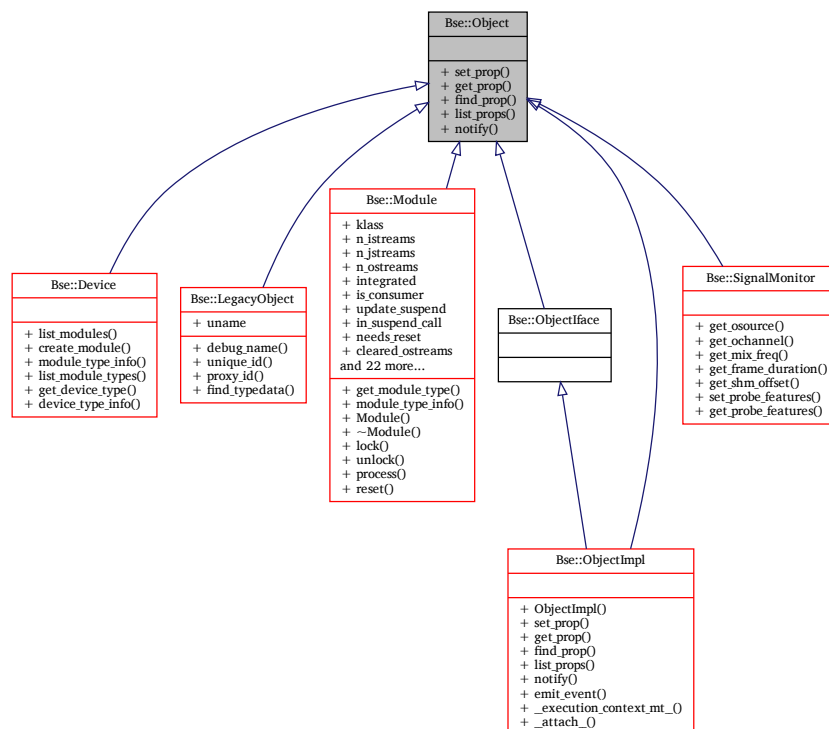
Detailed Description

A note description provides all needed details about a specific note. ”.
The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.65 Bse::Object Interface Reference

Base type for all new style C++ objects.
import "bseapi.idl";
Inheritance diagram for Bse::Object:



Public Member Functions

- bool `set_prop` (String name, Any value)
Set dynamic property name to value.
- Any `get_prop` (String name)
Get value of dynamic property name.
- StringSeq `find_prop` (String name)
Query key=value meta info for dynamic property name.
- StringSeq `list_props` ()
List names of all dynamic properties.
- void `notify` (String detail)
Emit notification event for object state or property changes.

Detailed Description

Base type for all new style C++ objects.

Member Function Documentation

`find_prop()`

```
StringSeq Bse::Object::find_prop (  
    String name )
```

Query key=value meta info for dynamic property name.

`get_prop()`

```
Any Bse::Object::get_prop (  
    String name )
```

Get value of dynamic property name.

`list_props()`

```
StringSeq Bse::Object::list_props ( )
```

List names of all dynamic properties.

`notify()`

```
void Bse::Object::notify (  
    String detail )
```

Emit notification event for object state or property changes.

`set_prop()`

```
bool Bse::Object::set_prop (  
    String name,  
    Any value )
```

Set dynamic property name to value.

The documentation for this interface was generated from the following file:

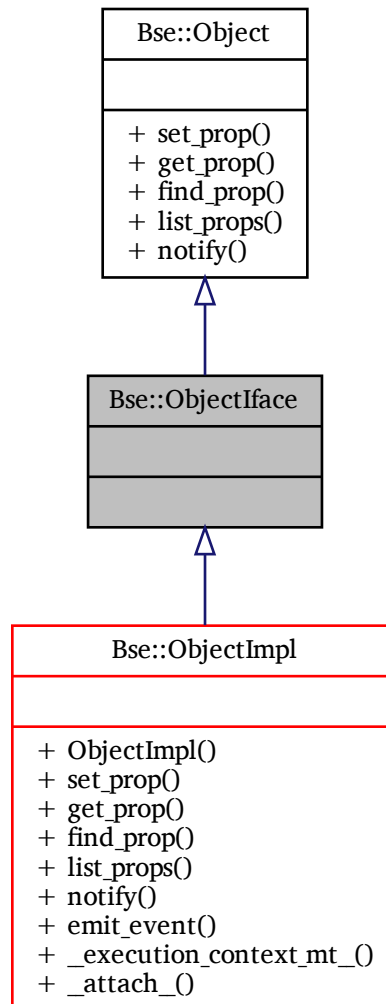
- [bse/bseapi.idl](#)

2.66 Bse::ObjectIface Class Reference

IDL interface class for [Bse::Object](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::ObjectIface:



Additional Inherited Members

Detailed Description

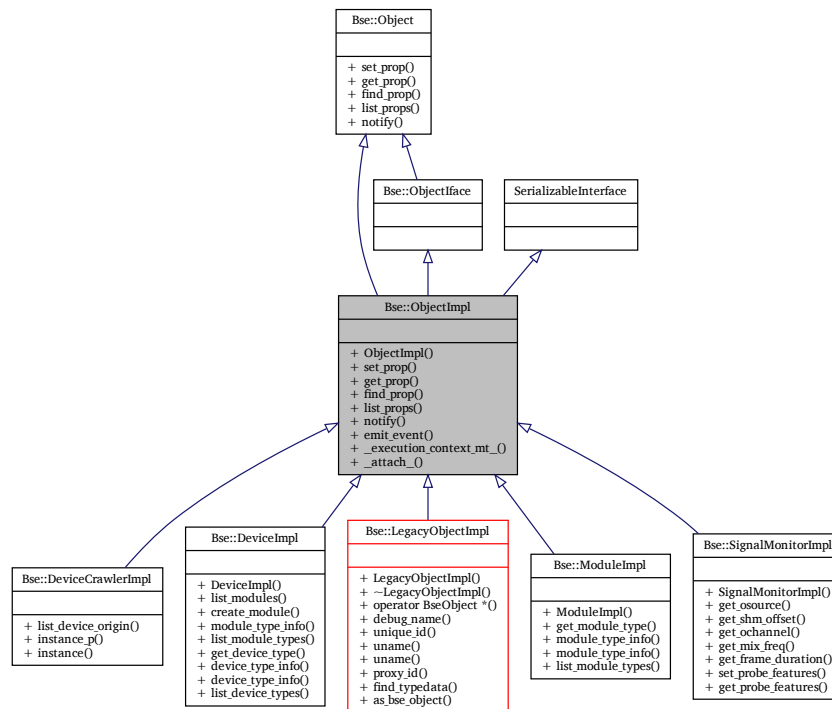
IDL interface class for [Bse::Object](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.67 Bse::ObjectImpl Class Reference

Inheritance diagram for Bse::ObjectImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

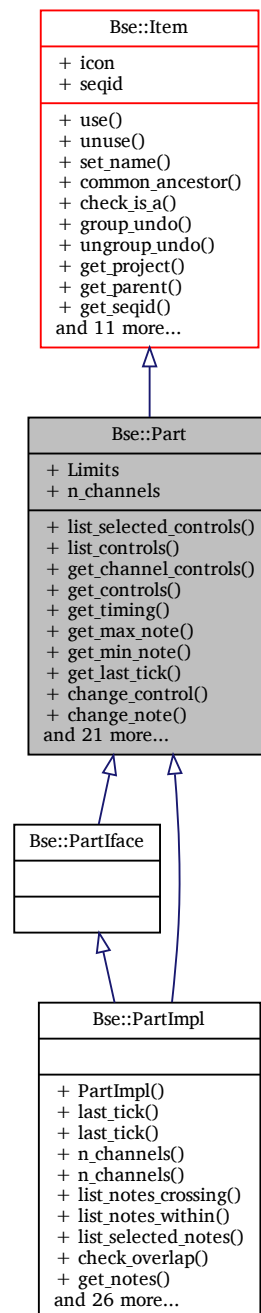
- bse/object.hh
- bse/object.cc

2.68 Bse::Part Interface Reference

Data interface for containment of piano notes and MIDI effects.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Part:



Public Member Functions

- **PartControlSeq list_selected_controls** (MidiSignal control_type)
List all currently selected control events of a specific type.
- **PartControlSeq list_controls** (int32 tick, int32 duration, MidiSignal control_type)
List all control events within a tick range.
- **PartControlSeq get_channel_controls** (int32 channel, int32 tick, int32 duration, MidiSignal control_type)
Retrieve all control events of a specific type within range of a channel.
- **PartControlSeq get_controls** (int32 tick, MidiSignal control_type)

- Retrieve all control events of a specific type at specified tick.*

 - **SongTiming get_timing** (int32 tick)
 - Retrieve song timing information at a specific tick.*
 - **int32 get_max_note** ()
 - Retrieve the maximum note supported in this part.*
 - **int32 get_min_note** ()
 - Retrieve the minimum note supported in this part.*
 - **int32 get_last_tick** ()
 - Retrieve the maximum tick any control or note plus duration spans.*
 - Error **change_control** (int32 id, int32 tick, MidiSignal control_type, float64 value)
 - Change an existing control event within a part.*
 - Error **change_note** (int32 id, int32 tick, int32 duration, int32 note, int32 fine_tune, float64 velocity)
 - Change an existing note within a part.*
 - Error **delete_event** (int32 id)
 - Delete an existing event from a part.*
 - **int32 insert_control** (int32 tick, MidiSignal control_type, float64 value)
 - Insert a new control event into a part.*
 - **int32 insert_note** (int32 channel, int32 tick, int32 duration, int32 note, int32 fine_tune, float64 velocity)
 - Insert a new note into a part.*
 - **int32 insert_note_auto** (int32 tick, int32 duration, int32 note, int32 fine_tune, float64 velocity)
 - Insert a new note into a part with automatic channel selection.*
 - bool **is_event_selected** (int32 id)
 - Check whether an event is selected.*
 - void **queue_controls** (int32 tick, int32 duration)
 - Queue updates for all control events and notes starting within the given range.*
 - void **queue_notes** (int32 tick, int32 duration, int32 min_note, int32 max_note)
 - Queue updates for all notes starting within the given rectangle.*
 - void **select_notes_exclusive** (int32 tick, int32 duration, int32 min_note, int32 max_note)
 - Select all notes within rectangle and deselect all others.*
 - void **select_controls_exclusive** (int32 tick, int32 duration, MidiSignal control_type)
 - Select all control events within range and deselect all others.*
 - void **select_notes** (int32 tick, int32 duration, int32 min_note, int32 max_note)
 - Select all notes within rectangle.*
 - void **select_event** (int32 id)
 - Select an existing event.*
 - void **select_controls** (int32 tick, int32 duration, MidiSignal control_type)
 - Select all control events within range.*
 - void **deselect_notes** (int32 tick, int32 duration, int32 min_note, int32 max_note)
 - Deselect all notes within rectangle.*
 - void **deselect_event** (int32 id)
 - Deselect an existing event.*
 - void **deselect_controls** (int32 tick, int32 duration, MidiSignal control_type)
 - Deselect all controls within given range.*
 - **PartNoteSeq list_notes_crossing** (int32 tick, int32 duration)
 - List all notes within or crossing a tick range.*
 - **PartNoteSeq list_notes_within** (int32 channel, int32 tick, int32 duration)
 - List all notes within a tick range.*
 - **PartNoteSeq list_selected_notes** ()
 - List all currently selected notes.*
 - **PartNoteSeq check_overlap** (int32 tick, int32 duration, int32 note)
 - Check whether a note would overlap with neighbours.*
 - **PartNoteSeq get_notes** (int32 tick, int32 note)
 - Retrieve all notes at a specific frequency or crossing a tick.*
 - **PartLinkSeq list_links** ()
 - List all places where parts are used (linked) from tracks, sorted by tick.*

Detailed Description

Data interface for containment of piano notes and MIDI effects.

Member Function Documentation

change_control()

```
Error Bse::Part::change_control (
    int32 id,
    int32 tick,
    MidiSignal control_type,
    float64 value )
```

Change an existing control event within a part.

change_note()

```
Error Bse::Part::change_note (
    int32 id,
    int32 tick,
    int32 duration,
    int32 note,
    int32 fine_tune,
    float64 velocity )
```

Change an existing note within a part.

check_overlap()

```
PartNoteSeq Bse::Part::check_overlap (
    int32 tick,
    int32 duration,
    int32 note )
```

Check whether a note would overlap with neighbours.

delete_event()

```
Error Bse::Part::delete_event (
    int32 id )
```

Delete an existing event from a part.

deselect_controls()

```
void Bse::Part::deselect_controls (
    int32 tick,
    int32 duration,
    MidiSignal control_type )
```

Deselect all controls within given range.

deselect_event()

```
void Bse::Part::deselect_event (
    int32 id )
```

Deselect an existing event.

deselect_notes()

```
void Bse::Part::deselect_notes (
    int32 tick,
    int32 duration,
    int32 min_note,
    int32 max_note )
```

Deselect all notes within rectangle.

get_channel_controls()

```
PartControlSeq Bse::Part::get_channel_controls (
    int32 channel,
    int32 tick,
    int32 duration,
    MidiSignal control_type )
```

Retrieve all control events of a specific type within range of a channel.

get_controls()

```
PartControlSeq Bse::Part::get_controls (
    int32 tick,
    MidiSignal control_type )
```

Retrieve all control events of a specific type at specified tick.

get_last_tick()

```
int32 Bse::Part::get_last_tick ( )
```

Retrieve the maximum tick any control or note plus duration spans.

get_max_note()

```
int32 Bse::Part::get_max_note ( )
```

Retrieve the maximum note supported in this part.

get_min_note()

```
int32 Bse::Part::get_min_note ( )
```

Retrieve the minimum note supported in this part.

get_notes()

```
PartNoteSeq Bse::Part::get_notes (
    int32 tick,
    int32 note )
```

Retrieve all notes at a specific frequency or crossing a tick.

get_timing()

```
SongTiming Bse::Part::get_timing (
    int32 tick )
```

Retrieve song timing information at a specific tick.

insert_control()

```
int32 Bse::Part::insert_control (
    int32 tick,
    MidiSignal control_type,
    float64 value )
```

Insert a new control event into a part.

insert_note()

```
int32 Bse::Part::insert_note (
    int32 channel,
    int32 tick,
    int32 duration,
    int32 note,
    int32 fine_tune,
    float64 velocity )
```

Insert a new note into a part.

insert_note_auto()

```
int32 Bse::Part::insert_note_auto (
    int32 tick,
    int32 duration,
    int32 note,
    int32 fine_tune,
    float64 velocity )
```

Insert a new note into a part with automatic channel selection.

is_event_selected()

```
bool Bse::Part::is_event_selected (
    int32 id )
```

Check whether an event is selected.

list_controls()

```
PartControlSeq Bse::Part::list_controls (
    int32 tick,
    int32 duration,
    MidiSignal control_type )
```

List all control events within a tick range.

list_links()

```
PartLinkSeq Bse::Part::list_links ( )
```

List all places where parts are used (linked) from tracks, sorted by tick.

list_notes_crossing()

```
PartNoteSeq Bse::Part::list_notes_crossing (
    int32 tick,
    int32 duration )
```

List all notes within or crossing a tick range.

list_notes_within()

```
PartNoteSeq Bse::Part::list_notes_within (
    int32 channel,
    int32 tick,
    int32 duration )
```

List all notes within a tick range.

list_selected_controls()

```
PartControlSeq Bse::Part::list_selected_controls (
    MidiSignal control_type )
```

List all currently selected control events of a specific type.

list_selected_notes()

```
PartNoteSeq Bse::Part::list_selected_notes ( )
```

List all currently selected notes.

queue_controls()

```
void Bse::Part::queue_controls (
    int32 tick,
    int32 duration )
```

Queue updates for all control events and notes starting within the given range.

queue_notes()

```
void Bse::Part::queue_notes (
    int32 tick,
    int32 duration,
    int32 min_note,
    int32 max_note )
```

Queue updates for all notes starting within the given rectangle.

select_controls()

```
void Bse::Part::select_controls (
    int32 tick,
    int32 duration,
    MidiSignal control_type )
```

Select all control events within range.

select_controls_exclusive()

```
void Bse::Part::select_controls_exclusive (
    int32 tick,
    int32 duration,
    MidiSignal control_type )
```

Select all control events within range and deselect all others.

select_event()

```
void Bse::Part::select_event (
    int32 id )
```

Select an existing event.

select_notes()

```
void Bse::Part::select_notes (
    int32 tick,
    int32 duration,
    int32 min_note,
    int32 max_note )
```

Select all notes within rectangle.

select_notes_exclusive()

```
void Bse::Part::select_notes_exclusive (
    int32 tick,
    int32 duration,
    int32 min_note,
    int32 max_note )
```

Select all notes within rectangle and deselect all others.

The documentation for this interface was generated from the following file:

- [bse/bseapi.idl](#)

2.69 Bse::PartControl Struct Reference

[Part](#) specific control event representation.

```
import "bseapi.idl";
```

Detailed Description

[Part](#) specific control event representation.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.70 Bse::PartControlSeq Struct Reference

A list of part control events.

```
import "bseapi.idl";
```

Detailed Description

A list of part control events.

The documentation for this struct was generated from the following file:

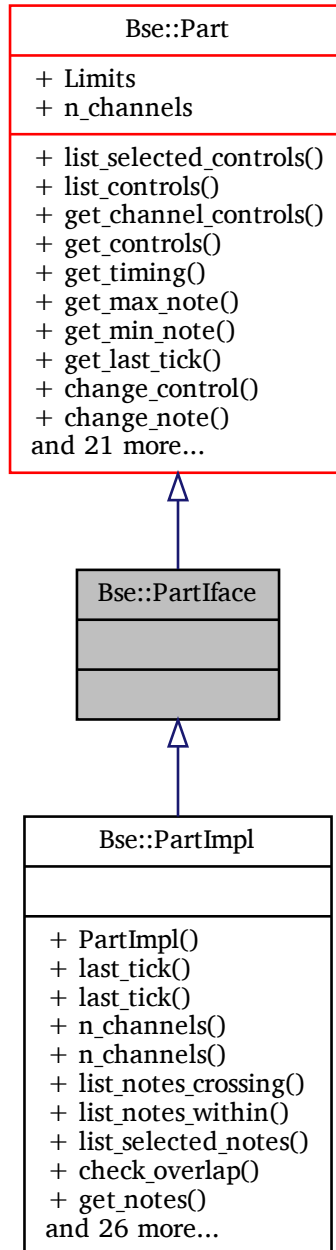
- [bse/bseapi.idl](#)

2.71 Bse::Partiface Class Reference

IDL interface class for [Bse::Part](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::Partiface:



Additional Inherited Members

Detailed Description

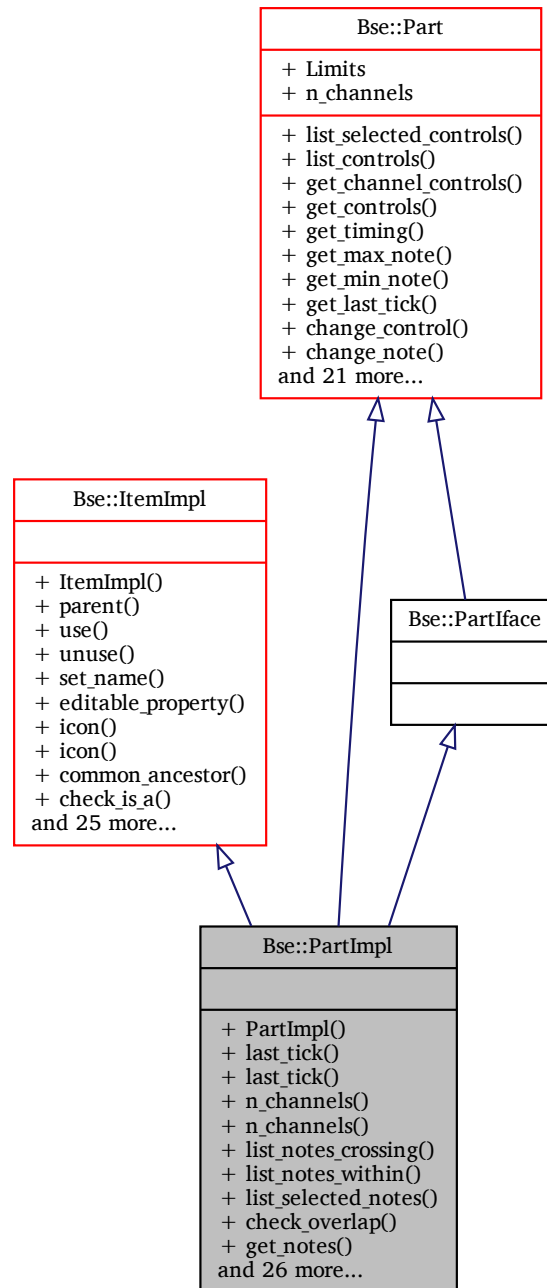
IDL interface class for [Bse::Part](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.72 Bse::PartImpl Class Reference

Inheritance diagram for Bse::PartImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- [bse/bsepart.hh](#)
- [bse/bsepart.cc](#)

2.73 Bse::PartLink Struct Reference

Record representing the use of a [Part](#) within a [Track](#) at a specific position.

```
import "bseapi.idl";
```

Detailed Description

Record representing the use of a [Part](#) within a [Track](#) at a specific position.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.74 Bse::PartLinkSeq Struct Reference

Sequence of [PartLink](#) records.

```
import "bseapi.idl";
```

Detailed Description

Sequence of [PartLink](#) records.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.75 Bse::PartNote Struct Reference

[Part](#) specific note event representation.

```
import "bseapi.idl";
```

Detailed Description

[Part](#) specific note event representation.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.76 Bse::PartNoteSeq Struct Reference

A list of part note events.

```
import "bseapi.idl";
```

Detailed Description

A list of part note events.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.77 Bse::PartSeq Struct Reference

A list of [Part](#) or derived types.

```
import "bseapi.idl";
```

Detailed Description

A list of [Part](#) or derived types.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.78 Bse::Pcg32Rng Class Reference

Pcg32Rng is a permutating linear congruential PRNG.

```
#include <randomhash.hh>
```

Public Member Functions

- `template<class SeedSeq >`
`Pcg32Rng (SeedSeq &seed_sequence)`
Initialize and seed from seed_sequence.
- `Pcg32Rng (uint64_t offset, uint64_t struct)`
Initialize and seed by seeking to position offset within stream sequence.
- `Pcg32Rng ()`
Initialize and seed the generator from a system specific nondeterministic random source.
- `void auto_seed ()`
Seed the generator from a system specific nondeterministic random source.
- `void seed (uint64_t offset, uint64_t struct)`
Seed by seeking to position offset within stream sequence.
- `template<class SeedSeq >`
`void seed (SeedSeq &seed_sequence)`
Seed the generator state from a seed_sequence.
- `uint32_t random ()`
Generate uniformly distributed 32 bit pseudo random number.

Detailed Description

Pcg32Rng is a permutating linear congruential PRNG.

At the core, this pseudo random number generator uses the well known linear congruential generator↔ : $6364136223846793005 * \text{accumulator} + 1442695040888963407 \bmod 2^{64}$. See also TAOCP by D. E. Knuth, section 3.3.4, table 1, line 26. For good statistical performance, the output function of the permuted congruential generator family is used as described on <http://www.pcg-random.org/>. Period length for this generator is 2^{64} , the specified seed *offset* chooses the position of the genrator and the seed *sequence* parameter can be used to choose from 2^{63} distinct random sequences.

Constructor & Destructor Documentation

Pcg32Rng() [1/3]

```
template<class SeedSeq >
Bse::Pcg32Rng::Pcg32Rng (
    SeedSeq & seed_sequence ) [inline], [explicit]
Initialize and seed from seed_sequence.
```

Pcg32Rng() [2/3]

```
Bse::Pcg32Rng::Pcg32Rng (
    uint64_t offset,
    uint64_t struct ) [explicit]
```

Initialize and seed by seeking to position *offset* within stream *sequence*.

Pcg32Rng() [3/3]

```
Bse::Pcg32Rng::Pcg32Rng ( ) [explicit]
```

Initialize and seed the generator from a system specific nondeterministic random source.

Member Function Documentation**auto_seed()**

```
void Bse::Pcg32Rng::auto_seed ( )
```

Seed the generator from a system specific nondeterministic random source.

random()

```
uint32_t Bse::Pcg32Rng::random ( ) [inline]
```

Generate uniformly distributed 32 bit pseudo random number.

seed() [1/2]

```
void Bse::Pcg32Rng::seed (
    uint64_t offset,
    uint64_t struct )
```

Seed by seeking to position *offset* within stream *sequence*.

seed() [2/2]

```
template<class SeedSeq >
void Bse::Pcg32Rng::seed (
    SeedSeq & seed_sequence ) [inline]
```

Seed the generator state from a *seed_sequence*.

The documentation for this class was generated from the following files:

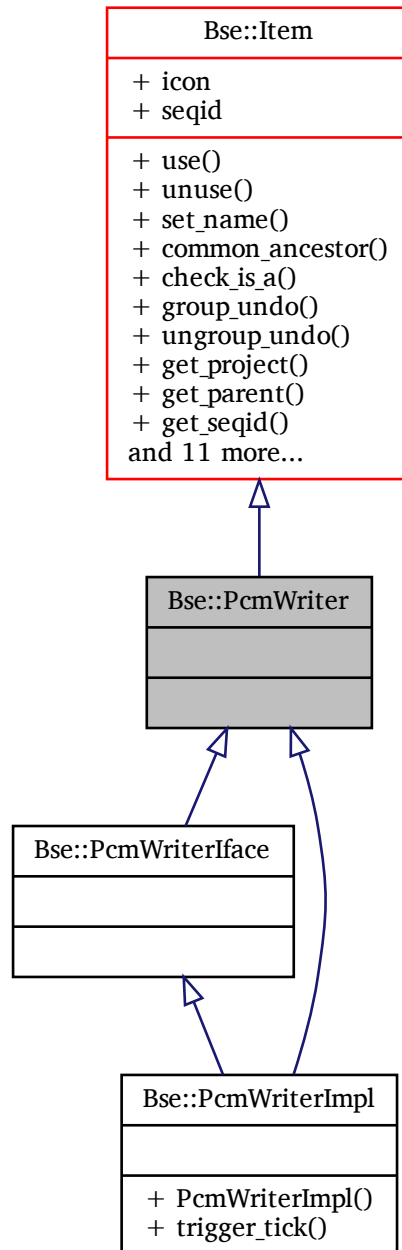
- bse/randomhash.hh
- bse/randomhash.cc

2.79 Bse::PcmWriter Interface Reference

Interface for writing PCM wave data.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::PcmWriter:



Additional Inherited Members

Detailed Description

Interface for writing PCM wave data.

The documentation for this interface was generated from the following file:

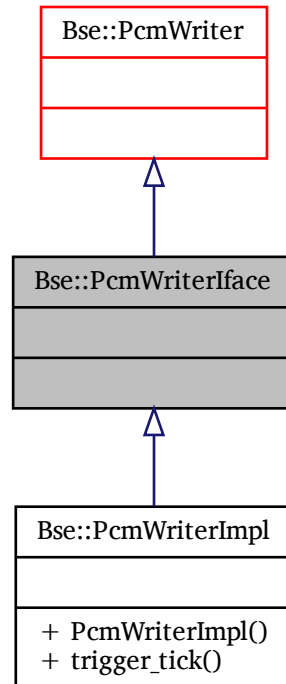
- [bse/bseapi.idl](#)

2.80 Bse::PcmWriterIface Class Reference

IDL interface class for [Bse::PcmWriter](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::PcmWriterIface:



Additional Inherited Members

Detailed Description

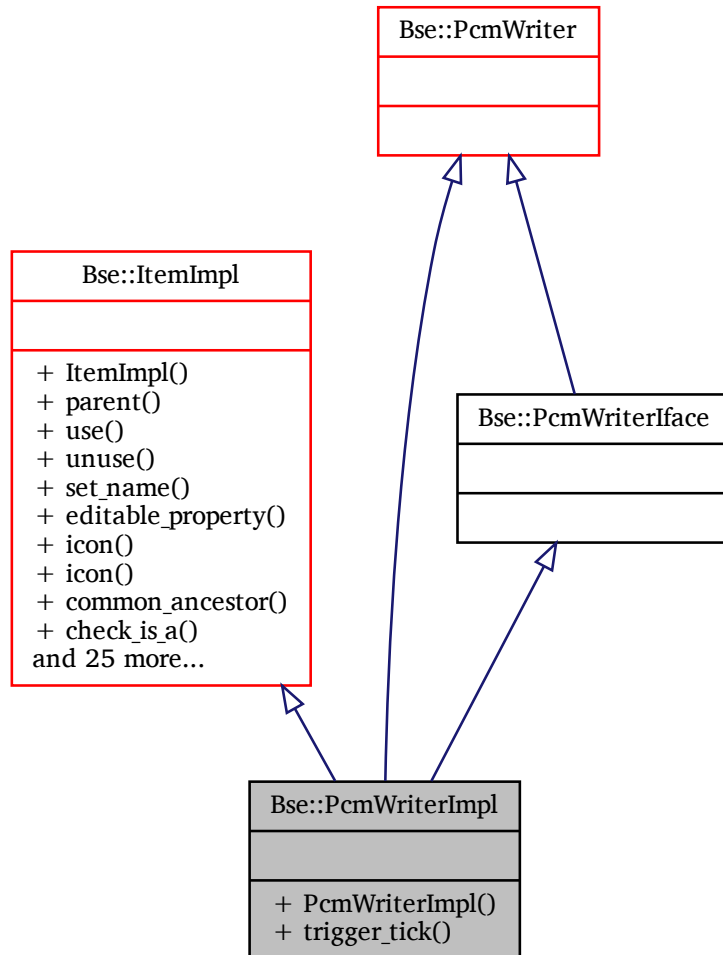
IDL interface class for [Bse::PcmWriter](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.81 Bse::PcmWriterImpl Class Reference

Inheritance diagram for Bse::PcmWriterImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- bse/bsepcmwriter.hh
- bse/bsepcmwriter.cc

2.82 Bse::PixelSeq Struct Reference

Representation of an image pixel sequence in ARGB format.
`import "bseapi.idl";`

Detailed Description

Representation of an image pixel sequence in ARGB format.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.83 Bse::ProbeFeatures Struct Reference

Bits representing a selection of probe sample data features.

```
import "bseapi.idl";
```

Detailed Description

Bits representing a selection of probe sample data features.

The documentation for this struct was generated from the following file:

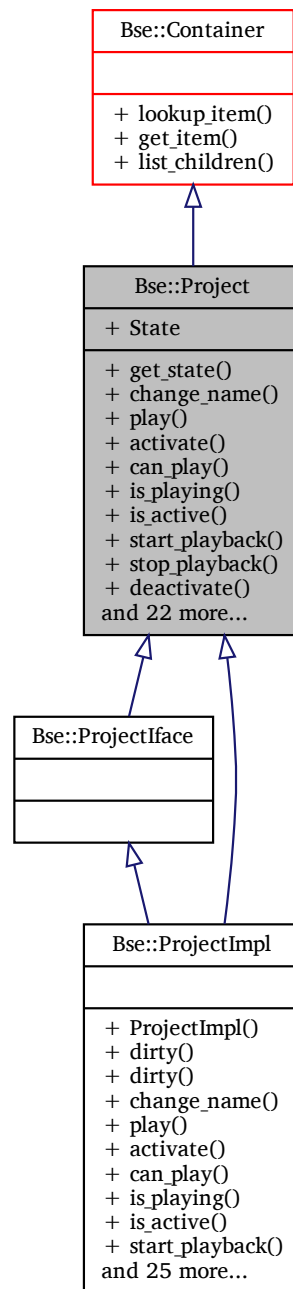
- [bse/bseapi.idl](#)

2.84 Bse::Project Interface Reference

Projects support loading, saving, playback and act as containers for all other sound objects.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Project:



Public Member Functions

- `ProjectState get_state ()`
Retrieve the current project activation/playback state.
- `void change_name (String name)`
Change a project name without recording undo steps.
- `Error play ()`
Activate a project and start project playback (an already playing project is first halted).
- `Error activate ()`

- Activate a project, precondition to start playback.*

 - bool `can_play` ()

Check whether project playback would makes sense.
- bool `is_playing` ()

Check whether a project is currently playing (song sequencing).
- bool `is_active` ()

Check whether a project is active (currently synthesizing).
- void `start_playback` ()

Start playback in an activated project.
- void `stop_playback` ()

Stop project playback.
- void `deactivate` ()

Deactivate the project, automatically stop playback.
- void `stop` ()

Stop project playback and deactivate project.
- void `auto_deactivate` (int32 msec_delay)

Automatically deactivate a few milliseconds after playback stopped.
- int32 `undo_depth` ()

Check whether a project can perform undo steps.
- void `undo` ()

Undo a previous operation in a project.
- int32 `redo_depth` ()

Get the number of times redo can be called on the project.
- void `redo` ()

Redo a previously undone operation in a project.
- void `clear_undo` ()

Delete all recorded undo or redo steps.
- void `clean_dirty` ()

Clear a project's dirty flags.
- bool `is_dirty` ()

Check whether a project needs saving.
- SuperSeq `get_supers` ()

*Retrieve all *Super* type objects of this project.*
- Error `store` (String file_name, bool self_contained)

Save project to file.
- Error `store_bse` (Container super, String file_name, bool self_contained)

Save effect or instrument.
- Song `create_song` (String name)

Create a song for this project.
- WaveRepo `get_wave_repo` ()

Retrieve the project's unique wave repository.
- SoundFontRepo `get_sound_font_repo` ()

Retrieve the project's unique sound font repository.
- CSynth `create_csynth` (String name)

Create a synthesizer network for this project.
- MidiSynth `create_midi_synth` (String name)

Create a MIDI synthesizer network for this project.
- MidiNotifier `get_midi_notifier` ()

Retrieve the project's midi notifier object.
- void `remove_snet` (SNet snet)

Remove an existing synthesizer network from this project.
- Error `restore_from_file` (String file_name)

Load a project from file.

- void `inject_midi_control` (`int32` midi_channel, `int32` midi_control, `float64` control_value)

Inject a MIDI control event into the project's MIDI receiver.

- Error `import_midi_file` (`String` file_name)

Import a song from a MIDI file.

Public Attributes

- group `State`

List unname paths for all items of a specified type within a project.

Detailed Description

Projects support loading, saving, playback and act as containers for all other sound objects.

Member Function Documentation

`activate()`

Error `Bse::Project::activate` ()

Activate a project, precondition to start playback.

`auto_deactivate()`

void `Bse::Project::auto_deactivate` (
 `int32` msec_delay)

Automatically deactivate a few milliseconds after playback stopped.

`can_play()`

bool `Bse::Project::can_play` ()

Check whether project playback would makes sense.

`change_name()`

void `Bse::Project::change_name` (
 `String` name)

Change a project name without recording undo steps.

`clean_dirty()`

void `Bse::Project::clean_dirty` ()

Clear a project's dirty flags.

`clear_undo()`

void `Bse::Project::clear_undo` ()

Delete all recorded undo or redo steps.

create_csynth()

```
CSynth Bse::Project::create_csynth (
    String name )
```

Create a synthsizer network for this project.

create_midi_synth()

```
MidiSynth Bse::Project::create_midi_synth (
    String name )
```

Create a MIDI synthesizer network for this project.

create_song()

```
Song Bse::Project::create_song (
    String name )
```

Create a song for this project.

deactivate()

```
void Bse::Project::deactivate ( )
```

Deactivate the project, automatically stop playback.

get_midi_notifier()

```
MidiNotifier Bse::Project::get_midi_notifier ( )
```

Retrieve the project's midi notifier object.

get_sound_font_repo()

```
SoundFontRepo Bse::Project::get_sound_font_repo ( )
```

Retrieve the project's unique sound font repository.

get_state()

```
ProjectState Bse::Project::get_state ( )
```

Retrieve the current project activation/playback state.

get_supers()

```
SuperSeq Bse::Project::get_supers ( )
```

Retrieve all [Super](#) type objects of this project.

get_wave_repo()

```
WaveRepo Bse::Project::get_wave_repo ( )
```

Retrieve the project's unique wave repository.

import_midi_file()

```
Error Bse::Project::import_midi_file (
    String file_name )
```

Import a song from a MIDI file.

inject_midi_control()

```
void Bse::Project::inject_midi_control (
    int32 midi_channel,
    int32 midi_control,
    float64 control_value )
```

Inject a MIDI control event into the project's MIDI receiver.

is_active()

```
bool Bse::Project::is_active ( )
```

Check whether a project is active (currently synthesizing).

is_dirty()

```
bool Bse::Project::is_dirty ( )
```

Check whether a project needs saving.

is_playing()

```
bool Bse::Project::is_playing ( )
```

Check whether a project is currently playing (song sequencing).

play()

```
Error Bse::Project::play ( )
```

Activate a project and start project playback (an already playing project is first halted).

redo()

```
void Bse::Project::redo ( )
```

Redo a previously undone operation in a project.

redo_depth()

```
int32 Bse::Project::redo_depth ( )
```

Get the number of times redo can be called on the project.

remove_snet()

```
void Bse::Project::remove_snet (
    SNet snet )
```

Remove an existing synthesizer network from this project.

restore_from_file()

```
Error Bse::Project::restore_from_file (
    String file_name )
```

Load a project from file.

start_playback()

```
void Bse::Project::start_playback ( )
```

Start playback in an activated project.

stop()

```
void Bse::Project::stop ( )
```

Stop project playback and deactivate project.

stop_playback()

```
void Bse::Project::stop_playback ( )
```

Stop project playback.

store()

```
Error Bse::Project::store (
    String file_name,
    bool self_contained )
```

Save project to file.

store_bse()

```
Error Bse::Project::store_bse (
    Container super,
    String file_name,
    bool self_contained )
```

Save effect or instrument.

undo()

```
void Bse::Project::undo ( )
```

Undo a previous operation in a project.

undo_depth()

```
int32 Bse::Project::undo_depth ( )
```

Check whether a project can perform undo steps.

Member Data Documentation

State

```
group Bse::Project::State
```

List unname paths for all items of a specified type within a project.

By their unname paths, items are uniquely identifiable within a project. Retrieve all items of a specific type within a project with matching unname. Save super objects of a project into a BSE file. If no [Super](#) is specified, the project itself is stored. The references to other objects (e.g. samples) can be stored by reference (`self_↔contained = false`) or embedded in the output file (`self_contained = true`).

The documentation for this interface was generated from the following file:

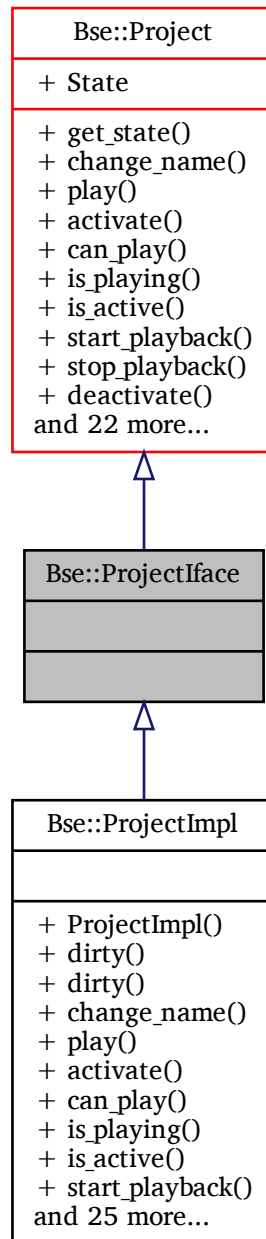
- [bse/bseapi.idl](#)

2.85 Bse::ProjectIface Class Reference

IDL interface class for [Bse::Project](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::ProjectIface:



Additional Inherited Members

Detailed Description

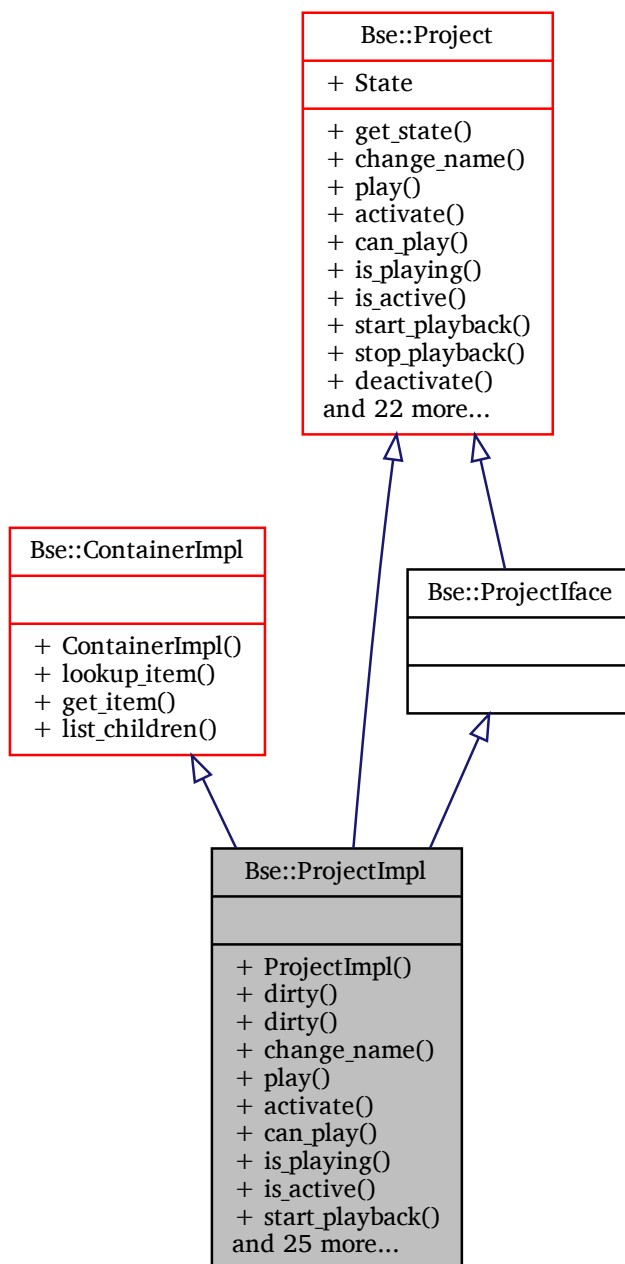
IDL interface class for [Bse::Project](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.86 Bse::ProjectImpl Class Reference

Inheritance diagram for Bse::ProjectImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- `bse/bseproject.hh`
- `bse/bseproject.cc`

2.87 Bse::PropertyCandidates Struct Reference

A list of items suitable to set as a specific property value.

```
import "bseapi.idl";
```

Detailed Description

A list of items suitable to set as a specific property value.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.88 Bse::Xms::SerializationNode::QueuedArgs Struct Reference

Helper for deferred `xml_reflink()` calls.

```
#include <serializable.hh>
```

Detailed Description

Helper for deferred `xml_reflink()` calls.

The documentation for this struct was generated from the following file:

- [bse/serializable.hh](#)

2.89 Bse::Xms::Reflink Class Reference

Representation for an object reference between [SerializableInterface](#) objects.

```
#include <serializable.hh>
```

Detailed Description

Representation for an object reference between [SerializableInterface](#) objects.

The documentation for this class was generated from the following files:

- [bse/serializable.hh](#)
- [bse/serializable.cc](#)

2.90 Bse::Resampler2 Class Reference

Interface for factor 2 resampling classes.

```
#include <bseresampler.hh>
```

Public Member Functions

- [Resampler2](#) (Mode mode, Precision precision, bool use_sse_if_available = true)
creates a resampler instance fulfilling a given specification
- void [process_block](#) (const **float** *input, **uint** n_input_samples, **float** *output)
resample a data block
- **uint** [order](#) () const
return FIR filter order
- **double** [delay](#) () const
Return the delay introduced by the resampler.
- void [reset](#) ()
clear internal history, reset resampler state to zero values
- bool [sse_enabled](#) () const
return whether the resampler is using sse optimized code

Static Public Member Functions

- static bool `sse_available ()`
returns true if an optimized SSE version of the Resampler is available
- static bool `test_filter_impl (bool verbose)`
test internal filter implementation
- static Precision `find_precision_for_bits (uint bits)`
finds a precision which is appropriate for at least the specified number of bits
- static const `char * precision_name (Precision precision)`
returns a human-readable name for a given precision

Detailed Description

Interface for factor 2 resampling classes.

Constructor & Destructor Documentation

Resampler2()

```
Resampler2::Resampler2 (
    Mode mode,
    Precision precision,
    bool use_sse_if_available = true )
```

creates a resampler instance fulfilling a given specification

Member Function Documentation

delay()

```
double Bse::Resampler2::delay ( ) const [inline]
```

Return the delay introduced by the resampler. This delay is guaranteed to be ≥ 0.0 , and for factor 2 resampling always a multiple of 0.5 (1.0 for upsampling). The return value can also be thought of as index into the output signal, where the first input sample can be found. Beware of fractional delays, for instance for downsampling, a `delay()` of 10.5 means that the first input sample would be found by interpolating `output[10]` and `output[11]`, and the second input sample equates `output[11]`.

find_precision_for_bits()

```
Resampler2::Precision Resampler2::find_precision_for_bits (
    uint bits ) [static]
```

finds a precision which is appropriate for at least the specified number of bits

order()

```
uint Bse::Resampler2::order ( ) const [inline]
```

return FIR filter order

precision_name()

```
const char * Resampler2::precision_name (
    Precision precision ) [static]
```

returns a human-readable name for a given precision

process_block()

```
void Bse::Resampler2::process_block (
    const float * input,
    uint n_input_samples,
    float * output ) [inline]
```

resample a data block

reset()

```
void Bse::Resampler2::reset ( ) [inline]
```

clear internal history, reset resampler state to zero values

sse_available()

```
bool Resampler2::sse_available ( ) [static]
```

returns true if an optimized SSE version of the Resampler is available

sse_enabled()

```
bool Bse::Resampler2::sse_enabled ( ) const [inline]
```

return whether the resampler is using sse optimized code

test_filter_impl()

```
bool Resampler2::test_filter_impl (
    bool verbose ) [static]
```

test internal filter implementation

The documentation for this class was generated from the following files:

- bse/bseresampler.hh
- bse/bseresampler.cc

2.91 Bse::SampleFileInfo Struct Reference

Structure containing meta data for multi wave samples.

```
import "bseapi.idl";
```

Detailed Description

Structure containing meta data for multi wave samples.

The documentation for this struct was generated from the following file:

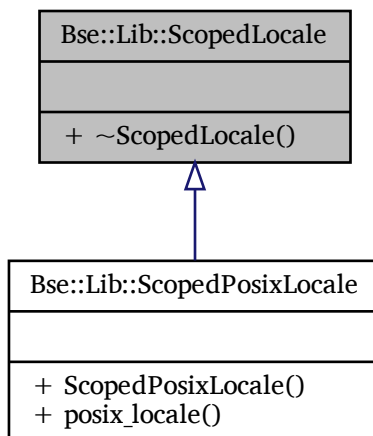
- [bse/bseapi.idl](#)

2.92 Bse::Lib::ScopedLocale Class Reference

Class to push a specific locale_t for the scope of its lifetime.

```
#include <formatter.hh>
```

Inheritance diagram for Bse::Lib::ScopedLocale:



Detailed Description

Class to push a specific locale_t for the scope of its lifetime.

The documentation for this class was generated from the following files:

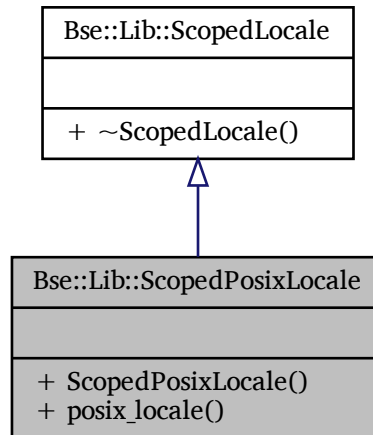
- bse/formatter.hh
- bse/formatter.cc

2.93 Bse::Lib::ScopedPosixLocale Class Reference

Class to push the POSIX/C locale_t (UTF-8) for the scope of its lifetime.

```
#include <formatter.hh>
```

Inheritance diagram for Bse::Lib::ScopedPosixLocale:



Static Public Member Functions

- static locale_t [posix_locale](#) ()
Retrieve the (UTF-8) POSIX/C locale_t.

Detailed Description

Class to push the POSIX/C locale_t (UTF-8) for the scope of its lifetime.

Member Function Documentation

posix_locale()

locale_t Bse::Lib::ScopedPosixLocale::posix_locale () [static]

Retrieve the (UTF-8) POSIX/C locale_t.

The documentation for this class was generated from the following files:

- bse/formatter.hh
- bse/formatter.cc

2.94 Bse::Sequencer Class Reference

Note and MIDI sequencer.

```
#include <bsesequencer.hh>
```

Detailed Description

Note and MIDI sequencer.

The sequencer processes notes from parts and MIDI input and generates events for the synthesis engine.

The documentation for this class was generated from the following files:

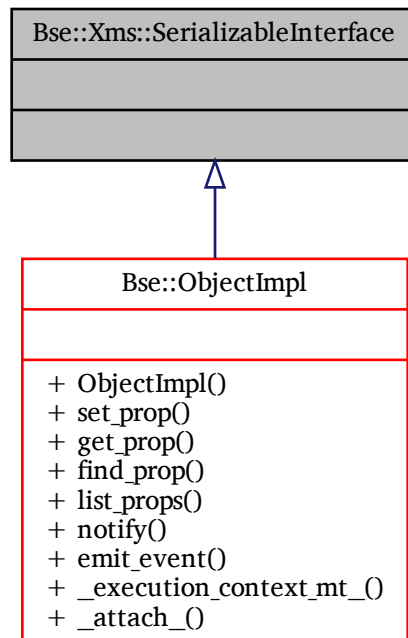
- bse/bsesequencer.hh
- bse/bsesequencer.cc

2.95 Bse::Xms::SerializableInterface Class Reference

Interface for serializable objects with [Reflink](#) support.

```
#include <serializable.hh>
```

Inheritance diagram for Bse::Xms::SerializableInterface:



Detailed Description

Interface for serializable objects with [Reflink](#) support.

The documentation for this class was generated from the following files:

- `bse/serializable.hh`
- `bse/serializable.cc`

2.96 Bse::Xms::SerializationField Class Reference

Reference to a storage attribute (or child node) for serialization.

```
#include <serializable.hh>
```

Public Member Functions

- `template<typename T, typename E = void>`
`void operator & (T &value)`
Serialization operator.
- `SerializationField & node ()`
Force storage into child node (not attribute)
- `bool as_node () const`
Retrive node() flag.
- `SerializationField & hex ()`

Hint for unsigned storage as hexadecimal.

- bool `as_hex ()` const
Retrive `hex()` flag.
- String `attribute ()` const
Associated attribute (or child node) name for serialization.
- `SerializationNode & serialization_node ()`
Associated storage node for serialization.

Detailed Description

Reference to a storage attribute (or child node) for serialization.

Member Function Documentation

as_hex()

`bool Bse::Xms::SerializationField::as_hex ()` const
Retrive `hex()` flag.

as_node()

`bool Bse::Xms::SerializationField::as_node ()` const
Retrive `node()` flag.

attribute()

`String Bse::Xms::SerializationField::attribute ()` const
Associated attribute (or child node) name for serialization.

hex()

`SerializationField & Bse::Xms::SerializationField::hex ()`
Hint for unsigned storage as hexadecimal.

node()

`SerializationField & Bse::Xms::SerializationField::node ()`
Force storage into child node (not attribute)

operator &()

```
template<typename T , typename E = void>
void Bse::Xms::SerializationField::operator& (
    T & value )
Serialization operator.
```

serialization_node()

`SerializationNode` & Bse::Xms::SerializationField::serialization_node ()

Associated storage node for serialization.

The documentation for this class was generated from the following files:

- bse/serializable.hh
- bse/serializable.cc

2.97 Bse::Xms::SerializationNode Class Reference

Representation of a storage node for serialization via `save()` and `load()`

```
#include <serializable.hh>
```

Classes

- struct `QueuedArgs`
Helper for deferred `xml_reflink()` calls.

Public Member Functions

- `SerializationNode` ()
Create a `SerializationNode` de-/serialization.
- `String name` () const
Tag name of this storage node.
- `operator bool` () const noexcept
Check if `this` exists as deserialization node.
- `SerializationNode create_child` (const `String` &childname)
Create a child node for nested hierarchies.
- `SerializationNode first_child` (const `String` &childname)
Access the first stored child node.
- `Children children` (const `String` &childname)
List all children by name.
- `SerializationField get` (const `String` &attrib)
Refer to a serialization field by name.
- `SerializationField operator[]` (const `String` &attrib)
Convenience operator alias for `get()`
- `bool has` (const `String` &attrib) const
Check for attribute or child presence.
- `bool loading` (const `String` &attrib) const
Combines `in_load()` with `has()` check.
- `bool in_load` () const
Return `true` during deserialization.
- `bool in_save` () const
Return `true` during serialization.
- `bool with_default` () const
Always `true` during `in_load()`, toggled during `in_save()`
- `void with_default` (bool dflt)
Toggle `with_default()` during `in_save()`
- `template<typename T >`
`Reflink & reflink` (T * &serializable_ptr)
Wrap serializable object pointers.
- `template<typename T >`
`void save_under` (const `String` &tag, T &object)

Serialize object into a child node.

- `template<typename T >`
`void load (T &object)`

Deserialize object via `xml_serialize()` and `xml_reflink()`

- `template<typename T >`
`void save (T &object)`

Serialize object via `xml_serialize()` and `xml_reflink()`

- `Bse::Error parse_xml (const String &root, const String &xmltext, String *ep = NULL)`

Parse `xmlstr`, returns error location and message in `ep`.

- `String write_xml (const String &root)`

Generate XML with toplevel tag `root` from this `SerializationNode`.

Static Public Attributes

- static constexpr const `char *const null_id`

String representing a serialized `nullptr`

Detailed Description

Representation of a storage node for serialization via `save()` and `load()`

Constructor & Destructor Documentation

SerializationNode()

`Bse::Xms::SerializationNode::SerializationNode ()` [explicit]

Create a `SerializationNode` de-/serialization.

Member Function Documentation

children()

`SerializationNode::Children Bse::Xms::SerializationNode::children (`
`const String & childname)`

List all children by name.

create_child()

`SerializationNode Bse::Xms::SerializationNode::create_child (`
`const String & childname)`

Create a child node for nested hierarchies.

first_child()

`SerializationNode Bse::Xms::SerializationNode::first_child (`
`const String & childname)`

Access the first stored child node.

get()

```
SerializationField Bse::Xms::SerializationNode::get (
    const String & attrib )
```

Refer to a serialization field by name.

has()

```
bool Bse::Xms::SerializationNode::has (
    const String & attrib ) const
```

Check for attribute or child presence.

in_load()

```
bool Bse::Xms::SerializationNode::in_load ( ) const
```

Return true during deserialization.

in_save()

```
bool Bse::Xms::SerializationNode::in_save ( ) const
```

Return true during serialization.

load()

```
template<typename T >
void Bse::Xms::SerializationNode::load (
    T & object )
```

Deserialize object via `xml_serialize()` and `xml_reflink()`

loading()

```
bool Bse::Xms::SerializationNode::loading (
    const String & attrib ) const
```

Combines `in_load()` with `has()` check.

name()

```
String Bse::Xms::SerializationNode::name ( ) const
```

Tag name of this storage node.

operator bool()

```
Bse::Xms::SerializationNode::operator bool ( ) const [explicit], [noexcept]
```

Check if this exists as deserialization node.

operator [] ()

```
SerializationField Bse::Xms::SerializationNode::operator[] (
    const String & attrib )
```

Convenience operator alias for `get()`

parse_xml()

```
Bse::Error Bse::Xms::SerializationNode::parse_xml (
    const String & root,
    const String & xmltext,
    String * ep = NULL )
```

Parse xmlstr, returns error location and message in ep.

reflink()

```
template<typename T >
Reflink & Bse::Xms::SerializationNode::reflink (
    T *& serializable_ptr )
```

Wrap serializable object pointers.

save()

```
template<typename T >
void Bse::Xms::SerializationNode::save (
    T & object )
```

Serialize object via `xml_serialize()` and `xml_reflink()`

save_under()

```
template<typename T >
void Bse::Xms::SerializationNode::save_under (
    const String & tag,
    T & object )
```

Serialize object into a child node.

with_default() [1/2]

```
bool Bse::Xms::SerializationNode::with_default ( ) const
```

Always true during `in_load()`, toggled during `in_save()`

with_default() [2/2]

```
void Bse::Xms::SerializationNode::with_default (
    bool dflt )
```

Toggle `with_default()` during `in_save()`

write_xml()

```
String Bse::Xms::SerializationNode::write_xml (
    const String & root )
```

Generate XML with toplevel tag root from this `SerializationNode`.

Member Data Documentation

null_id

```
constexpr const char* const Bse::Xms::SerializationNode::null_id [static]
```

String representing a serialized nullptr

The documentation for this class was generated from the following files:

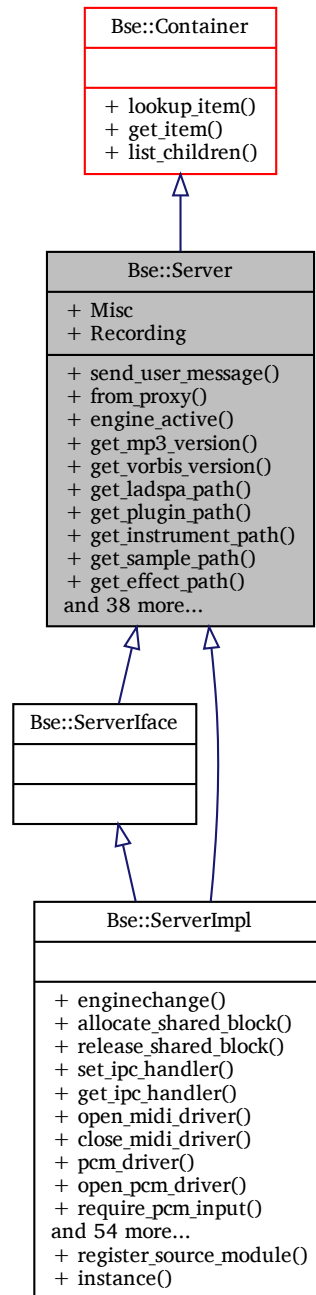
- bse/serializable.hh
- bse/serializable.cc

2.98 Bse::Server Interface Reference

Main [Bse](#) remote origin object.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Server:



Public Member Functions

- void `send_user_message` (`UserMessage` umsg)
Send a user messages from BSE.
- LegacyObject `from_proxy` (`int64` proxyid)
Find an Object from its associated BseObject proxy id.
- bool `engine_active` ()
Retrieve DSP engine activateion state, see also: "enginechange" Event.
- String `get_mp3_version` ()

- Retrieve BSE MP3 handler version.*

 - [String get_vorbis_version \(\)](#)
- Retrieve BSE Vorbis handler version.*

 - [String get_ladspa_path \(\)](#)
- Retrieve ladspa search path.*

 - [String get_plugin_path \(\)](#)
- Retrieve plugin search path.*

 - [String get_instrument_path \(\)](#)
- Retrieve instrument search path.*

 - [String get_sample_path \(\)](#)
- Retrieve sample search path.*

 - [String get_effect_path \(\)](#)
- Retrieve effect search path.*

 - [String get_demo_path \(\)](#)
- Retrieve demo search path.*

 - [String get_custom_instrument_dir \(\)](#)
- Retrieve user specific instruments directory.*

 - [String get_custom_effect_dir \(\)](#)
- Retrieve user specific effects directory.*

 - [String get_version \(\)](#)
- Retrieve BSE version.*

 - [void purge_stale_cachedirs \(\)](#)
- Purge stale directories from past runtimes.*

 - [void register_ladspa_plugins \(\)](#)
- Register LADSPA (Linux Audio Developer's Simple Plugin API) plugins asynchronously, DEPRECATED.*

 - [void register_core_plugins \(\)](#)
- Register core plugins asynchronously, DEPRECATED.*

 - [void load_assets \(\)](#)
- Register plugins and scripts immediately.*

 - [bool can_load \(String file_name\)](#)
- Check whether a loader can be found for a wave file.*

 - [void start_recording \(String wave_file, float64 n_seconds\)](#)
- Start recording to a WAV file.*

 - [Project create_project \(String project_name\)](#)
- Create a new project (name is modified to be unique if necessary).*

 - [Project last_project \(\)](#)
- Retrieve the last created project.*

 - [void destroy_project \(Project project\)](#)
- Destroy a previously created new project.*

 - [AuxDataSeq list_module_types \(\)](#)
- A list of Source type names for create_source().*

 - [AuxData find_module_type \(String module_type\)](#)
- Retrieve info about a Source type names.*

 - [Icon module_type_icon \(String module_type\)](#)
- Retrieve the icon associated with a module type.*

 - [SampleFileInfo sample_file_info \(String file_name\)](#)
- Load sample file info from file.*

 - [void broadcast_shm_fragments \(ShmFragmentSeq plan, int32 interval_ms\)](#)
- Broadcast shared memory fragments to the current Jsonipc connection.*

 - [SharedMemory get_shared_memory \(\)](#)
- Retrieve global SharedMemory information.*

 - [Configuration get_config_defaults \(\)](#)

- Retrieve Bse::Configuration setting defaults.*
- void `set_config` (Configuration configuration)
 - Assign updated Bse::Configuration settings.*
- Configuration `get_config` ()
 - Retrieve Bse::Configuration settings.*
- bool `locked_config` ()
 - Returns whether Configuration is in use and locked against modifications.*
- `DriverEntrySeq list_pcm_drivers` ()
 - List available drivers for PCM input/output handling.*
- `DriverEntrySeq list_midi_drivers` ()
 - List available drivers for MIDI input/output handling.*
- `NoteDescription note_describe` (MusicalTuning musical_tuning, `int32` note, `int32` fine_tune)
 - Describe a note, providing information about its octave, semitone, frequency, etc.*
- `NoteDescription note_construct` (MusicalTuning musical_tuning, `int32` semitone, `int32` octave, `int32` fine_tune)
 - Describe a note, given its semitone, octave and fine tune.*
- `NoteDescription note_from_string` (MusicalTuning musical_tuning, `String` name)
 - Describe a note, given its name and octave offset.*
- `int32 note_from_freq` (MusicalTuning musical_tuning, float64 frequency)
 - Retrieve the note of a certain frequency.*
- float64 `note_to_freq` (MusicalTuning musical_tuning, `int32` note, `int32` fine_tune)
 - Retrieve the frequency of a certain note.*
- `CategorySeq category_match_typed` (`String` pattern, `String` type_name)
 - List BSE categories according to a pattern and type match.*
- `CategorySeq category_match` (`String` pattern)
 - List BSE categories according to a pattern match.*
- `int64 tick_stamp_from_sysime` (`int64` systime_uses)
 - Helper for Wave PCM positioning.*
- `int32 test_counter_inc_fetch` ()
 - Testing, increment and return the resulting test counter value.*

Detailed Description

Main Bse remote origin object.

The `Bse::Server` object controls the main BSE thread and keeps track of all objects used in the BSE context.

Events:

- **enginechange** - A notification event for DSP engine changes, the event field *active* contains the activation state.

Member Function Documentation

`broadcast_shm_fragments()`

```
void Bse::Server::broadcast_shm_fragments (
    ShmFragmentSeq plan,
    int32 interval_ms )
```

Broadcast shared memory fragments to the current Jsonipc connection.

can_load()

```
bool Bse::Server::can_load (
    String file_name )
```

Check whether a loader can be found for a wave file.

category_match()

```
CategorySeq Bse::Server::category_match (
    String pattern )
```

List BSE categories according to a pattern match.

category_match_typed()

```
CategorySeq Bse::Server::category_match_typed (
    String pattern,
    String type_name )
```

List BSE categories according to a pattern and type match.

create_project()

```
Project Bse::Server::create_project (
    String project_name )
```

Create a new project (name is modified to be unique if necessary).

destroy_project()

```
void Bse::Server::destroy_project (
    Project project )
```

Destroy a previously created new project.

engine_active()

```
bool Bse::Server::engine_active ( )
```

Retrieve DSP engine activateion state, see also: "enginechange" Event.

find_module_type()

```
AuxData Bse::Server::find_module_type (
    String module_type )
```

Retrieve info about a [Source](#) type names.

from_proxy()

```
LegacyObject Bse::Server::from_proxy (
    int64 proxyid )
```

Find an [Object](#) from its associated BseObject proxy id.

get_config()

```
Configuration Bse::Server::get_config ( )
```

Retrieve Bse::Configuration settings.

get_config_defaults()

Configuration Bse::Server::get_config_defaults ()
Retrieve Bse::Configuration setting defaults.

get_custom_effect_dir()

String Bse::Server::get_custom_effect_dir ()
Retrieve user specific effects directory.

get_custom_instrument_dir()

String Bse::Server::get_custom_instrument_dir ()
Retrieve user specific instruments directory.

get_demo_path()

String Bse::Server::get_demo_path ()
Retrieve demo search path.

get_effect_path()

String Bse::Server::get_effect_path ()
Retrieve effect search path.

get_instrument_path()

String Bse::Server::get_instrument_path ()
Retrieve instrument search path.

get_ladspa_path()

String Bse::Server::get_ladspa_path ()
Retrieve ladspa search path.

get_mp3_version()

String Bse::Server::get_mp3_version ()
Retrieve BSE MP3 handler version.

get_plugin_path()

String Bse::Server::get_plugin_path ()
Retrieve plugin search path.

get_sample_path()

String Bse::Server::get_sample_path ()
Retrieve sample search path.

get_shared_memory()

`SharedMemory` Bse::Server::get_shared_memory ()
Retrieve global `SharedMemory` information.

get_version()

`String` Bse::Server::get_version ()
Retrieve BSE version.

get_vorbis_version()

`String` Bse::Server::get_vorbis_version ()
Retrieve BSE Vorbis handler version.

last_project()

`Project` Bse::Server::last_project ()
Retrieve the last created project.

list_midi_drivers()

`DriverEntrySeq` Bse::Server::list_midi_drivers ()
List available drivers for MIDI input/output handling.

list_module_types()

`AuxDataSeq` Bse::Server::list_module_types ()
A list of `Source` type names for `create_source()`.

list_pcm_drivers()

`DriverEntrySeq` Bse::Server::list_pcm_drivers ()
List available drivers for PCM input/output handling.

load_assets()

`void` Bse::Server::load_assets ()
Register plugins and scripts immediately.

locked_config()

`bool` Bse::Server::locked_config ()
Returns whether Configuration is in use and locked against modifications.

module_type_icon()

`Icon` Bse::Server::module_type_icon (
 `String` *module_type*)
Retrieve the icon associated with a module type.

note_construct()

```
NoteDescription Bse::Server::note_construct (
    MusicalTuning musical_tuning,
    int32 semitone,
    int32 octave,
    int32 fine_tune )
```

Describe a note, given its semitone, octave and fine tune.

note_describe()

```
NoteDescription Bse::Server::note_describe (
    MusicalTuning musical_tuning,
    int32 note,
    int32 fine_tune )
```

Describe a note, providing information about its octave, semitone, frequency, etc.

note_from_freq()

```
int32 Bse::Server::note_from_freq (
    MusicalTuning musical_tuning,
    float64 frequency )
```

Retrieve the note of a certain frequency.

note_from_string()

```
NoteDescription Bse::Server::note_from_string (
    MusicalTuning musical_tuning,
    String name )
```

Describe a note, given its name and octave offset.

note_to_freq()

```
float64 Bse::Server::note_to_freq (
    MusicalTuning musical_tuning,
    int32 note,
    int32 fine_tune )
```

Retrieve the frequency of a certain note.

purge_stale_cachedirs()

```
void Bse::Server::purge_stale_cachedirs ( )
```

Purge stale directories from past runtimes.

register_core_plugins()

```
void Bse::Server::register_core_plugins ( )
```

Register core plugins asynchronously, DEPRECATED.

register_ladspa_plugins()

```
void Bse::Server::register_ladspa_plugins ( )
```

Register LADSPA (Linux Audio Developer's Simple Plugin API) plugins asynchronously, DEPRECATED.

sample_file_info()

```
SampleFileInfo Bse::Server::sample_file_info (
    String file_name )
```

Load sample file info from file.

send_user_message()

```
void Bse::Server::send_user_message (
    UserMessage umsg )
```

Send a user messages from BSE.

set_config()

```
void Bse::Server::set_config (
    Configuration configuration )
```

Assign updated Bse::Configuration settings.

start_recording()

```
void Bse::Server::start_recording (
    String wave_file,
    float64 n_seconds )
```

Start recording to a WAV file.

test_counter_inc_fetch()

```
int32 Bse::Server::test_counter_inc_fetch ( )
```

Testing, increment and return the resulting test counter value.

tick_stamp_from_systemtime()

```
int64 Bse::Server::tick_stamp_from_systemtime (
    int64 systime_usecs )
```

Helper for [Wave](#) PCM positioning.

The documentation for this interface was generated from the following file:

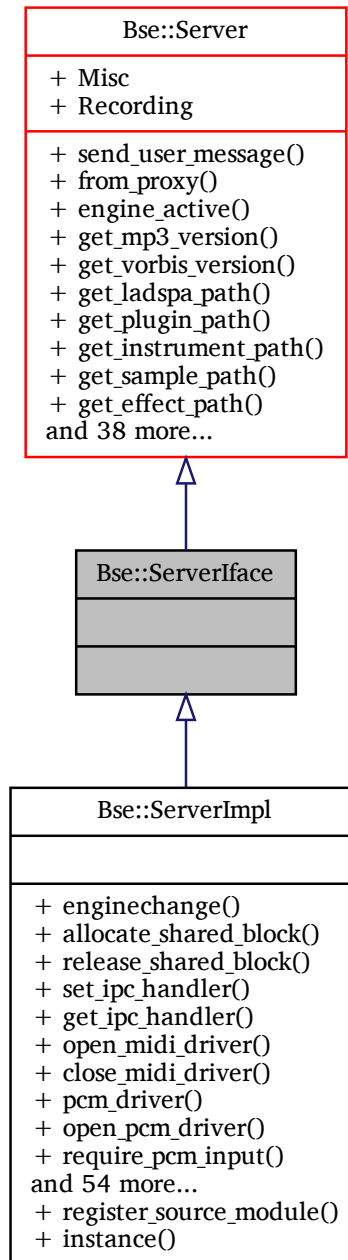
- [bse/bseapi.idl](#)

2.99 Bse::ServerIface Class Reference

IDL interface class for [Bse::Server](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::ServerIface:



Additional Inherited Members

Detailed Description

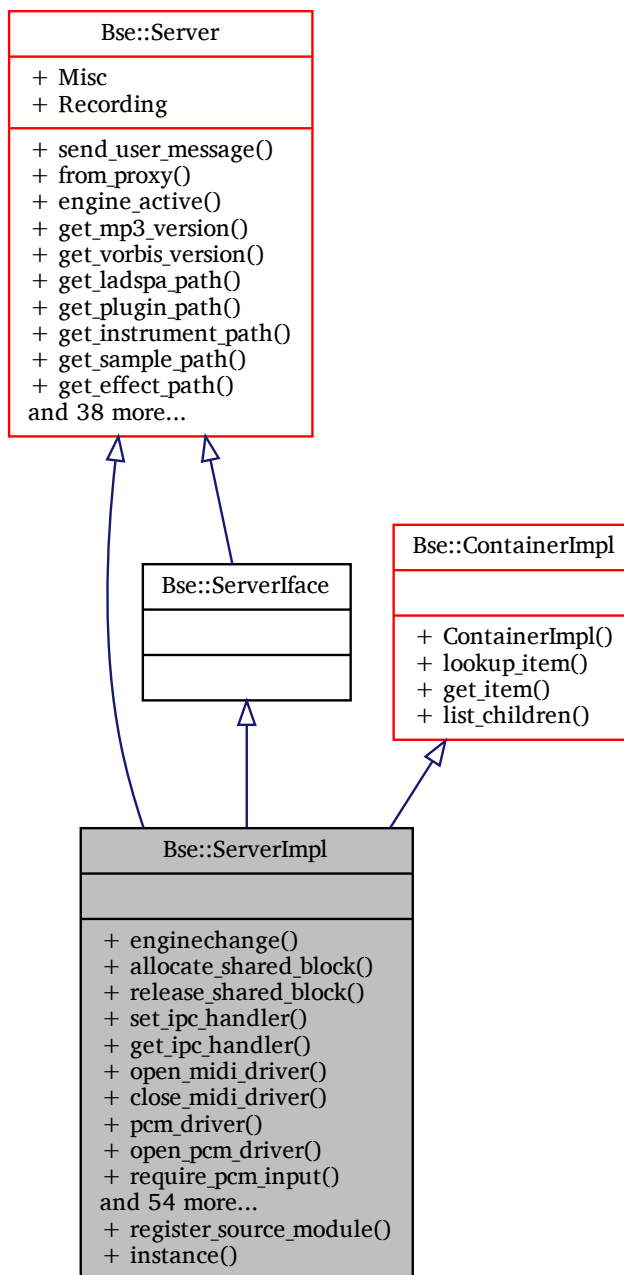
IDL interface class for [Bse::Server](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.100 Bse::ServerImpl Class Reference

Inheritance diagram for Bse::ServerImpl:



Static Public Member Functions

- static void `register_source_module` (const `String` &type, const `String` &title, const `String` &tags, const `uint8` *pixstream)

Register a synthesis module type at program startup.

Additional Inherited Members

Detailed Description

Member Function Documentation

register_source_module()

```
void Bse::ServerImpl::register_source_module (
    const String & type,
    const String & title,
    const String & tags,
    const uint8 * pixstream ) [static]
```

Register a synthesis module type at program startup.

The documentation for this class was generated from the following files:

- bse/bseserver.hh
- bse/bseserver.cc

2.101 SfiRecFields Struct Reference

Pointer sized integer object handle.

```
#include <sfitypes.hh>
```

Detailed Description

Pointer sized integer object handle.

The documentation for this struct was generated from the following file:

- bse/sfitypes.hh

2.102 Bse::SHA3_224 Struct Reference

[SHA3_224](#) - 224 Bit digest generation.

```
#include <randomhash.hh>
```

Public Member Functions

- [SHA3_224](#) ()
Create context to calculate a 224 bit SHA3 hash digest.
- void [reset](#) ()
Reset state to feed and retrieve a new hash value.
- void [update](#) (const **uint8_t** *data, **size_t** length)
Feed data to be hashed.
- void [digest](#) (**uint8_t** hashvalue[28])
Retrieve the resulting hash value.

Detailed Description

[SHA3_224](#) - 224 Bit digest generation.

This class implements the SHA3 hash function to create 224 Bit digests, see FIPS 202 [Division \(2014\)](#) .

Constructor & Destructor Documentation

SHA3_224()

```
Bse::SHA3_224::SHA3_224 ( )
```

Create context to calculate a 224 bit SHA3 hash digest.

Member Function Documentation

digest()

```
void Bse::SHA3_224::digest (
    uint8_t hashvalue[28] )
```

Retrieve the resulting hash value.

reset()

```
void Bse::SHA3_224::reset ( )
```

Reset state to feed and retrieve a new hash value.

update()

```
void Bse::SHA3_224::update (
    const uint8_t * data,
    size_t length )
```

Feed data to be hashed.

The documentation for this struct was generated from the following files:

- bse/randomhash.hh
- bse/randomhash.cc

2.103 Bse::SHA3_256 Struct Reference

[SHA3_256](#) - 256 Bit digest generation.

```
#include <randomhash.hh>
```

Public Member Functions

- [SHA3_256](#) ()
Create context to calculate a 256 bit SHA3 hash digest.
- void [reset](#) ()
Reset state to feed and retrieve a new hash value.
- void [update](#) (const **uint8_t** *data, **size_t** length)
Feed data to be hashed.
- void [digest](#) (**uint8_t** hashvalue[32])
Retrieve the resulting hash value.

Detailed Description

[SHA3_256](#) - 256 Bit digest generation.

This class implements the SHA3 hash function to create 256 Bit digests, see FIPS 202 [Division \(2014\)](#) .

Constructor & Destructor Documentation

SHA3_256()

```
Bse::SHA3_256::SHA3_256 ( )
```

Create context to calculate a 256 bit SHA3 hash digest.

Member Function Documentation

digest()

```
void Bse::SHA3_256::digest (
    uint8_t hashvalue[32] )
```

Retrieve the resulting hash value.

reset()

```
void Bse::SHA3_256::reset ( )
```

Reset state to feed and retrieve a new hash value.

update()

```
void Bse::SHA3_256::update (
    const uint8_t * data,
    size_t length )
```

Feed data to be hashed.

The documentation for this struct was generated from the following files:

- bse/randomhash.hh
- bse/randomhash.cc

2.104 Bse::SHA3_384 Struct Reference

[SHA3_384](#) - 384 Bit digest generation.

```
#include <randomhash.hh>
```

Public Member Functions

- [SHA3_384](#) ()
Create context to calculate a 384 bit SHA3 hash digest.
- void [reset](#) ()
Reset state to feed and retrieve a new hash value.
- void [update](#) (const **uint8_t** *data, **size_t** length)
Feed data to be hashed.
- void [digest](#) (**uint8_t** hashvalue[48])
Retrieve the resulting hash value.

Detailed Description

[SHA3_384](#) - 384 Bit digest generation.

This class implements the SHA3 hash function to create 384 Bit digests, see FIPS 202 [Division \(2014\)](#) .

Constructor & Destructor Documentation

SHA3_384()

```
Bse::SHA3_384::SHA3_384 ( )
```

Create context to calculate a 384 bit SHA3 hash digest.

Member Function Documentation**digest()**

```
void Bse::SHA3_384::digest (
    uint8_t hashvalue[48] )
```

Retrieve the resulting hash value.

reset()

```
void Bse::SHA3_384::reset ( )
```

Reset state to feed and retrieve a new hash value.

update()

```
void Bse::SHA3_384::update (
    const uint8_t * data,
    size_t length )
```

Feed data to be hashed.

The documentation for this struct was generated from the following files:

- bse/randomhash.hh
- bse/randomhash.cc

2.105 Bse::SHA3_512 Struct Reference

[SHA3_512](#) - 512 Bit digest generation.

```
#include <randomhash.hh>
```

Public Member Functions

- [SHA3_512 \(\)](#)
Create context to calculate a 512 bit SHA3 hash digest.
- void [reset \(\)](#)
Reset state to feed and retrieve a new hash value.
- void [update](#) (const **uint8_t** *data, **size_t** length)
Feed data to be hashed.
- void [digest](#) (**uint8_t** hashvalue[64])
Retrieve the resulting hash value.

Detailed Description

[SHA3_512](#) - 512 Bit digest generation.

This class implements the SHA3 hash function to create 512 Bit digests, see FIPS 202 [Division \(2014\)](#) .

Constructor & Destructor Documentation

SHA3_512()

```
Bse::SHA3_512::SHA3_512 ( )
```

Create context to calculate a 512 bit SHA3 hash digest.

Member Function Documentation

digest()

```
void Bse::SHA3_512::digest (
    uint8_t hashvalue[64] )
```

Retrieve the resulting hash value.

reset()

```
void Bse::SHA3_512::reset ( )
```

Reset state to feed and retrieve a new hash value.

update()

```
void Bse::SHA3_512::update (
    const uint8_t * data,
    size_t length )
```

Feed data to be hashed.

The documentation for this struct was generated from the following files:

- bse/randomhash.hh
- bse/randomhash.cc

2.106 Bse::SHAKE128 Struct Reference

[SHAKE128](#) - 128 Bit extendable output digest generation.

```
#include <randomhash.hh>
```

Public Member Functions

- [SHAKE128](#) ()
Create context to calculate an unbounded [SHAKE128](#) hash digest.
- void [reset](#) ()
Reset state to feed and retrieve a new hash value.
- void [update](#) (const **uint8_t** *data, **size_t** length)
Feed data to be hashed.
- void [squeeze_digest](#) (**uint8_t** *hashvalues, **size_t** n)
Retrieve an arbitrary number of hash value bytes.

Detailed Description

[SHAKE128](#) - 128 Bit extendable output digest generation.

This class implements the SHA3 extendable output hash function with 128 bit security strength, see [FIPS 202 Division \(2014\)](#) .

Constructor & Destructor Documentation

SHAKE128()

```
Bse::SHAKE128::SHAKE128 ( )
```

Create context to calculate an unbounded [SHAKE128](#) hash digest.

Member Function Documentation

reset()

```
void Bse::SHAKE128::reset ( )
```

Reset state to feed and retrieve a new hash value.

squeeze_digest()

```
void Bse::SHAKE128::squeeze_digest (
    uint8_t * hashvalues,
    size_t n )
```

Retrieve an arbitrary number of hash value bytes.

update()

```
void Bse::SHAKE128::update (
    const uint8_t * data,
    size_t length )
```

Feed data to be hashed.

The documentation for this struct was generated from the following files:

- bse/randomhash.hh
- bse/randomhash.cc

2.107 Bse::SHAKE256 Struct Reference

[SHAKE256](#) - 256 Bit extendable output digest generation.

```
#include <randomhash.hh>
```

Public Member Functions

- [SHAKE256](#) ()
Create context to calculate an unbounded [SHAKE256](#) hash digest.
- void [reset](#) ()
Reset state to feed and retrieve a new hash value.
- void [update](#) (const **uint8_t** *data, **size_t** length)
Feed data to be hashed.
- void [squeeze_digest](#) (**uint8_t** *hashvalues, **size_t** n)
Retrieve an arbitrary number of hash value bytes.

Detailed Description

[SHAKE256](#) - 256 Bit extendable output digest generation.

This class implements the SHA3 extendable output hash function with 256 bit security strength, see [FIPS 202 Division \(2014\)](#) .

Constructor & Destructor Documentation

SHAKE256()

```
Bse::SHAKE256::SHAKE256 ( )
```

Create context to calculate an unbounded [SHAKE256](#) hash digest.

Member Function Documentation

reset()

```
void Bse::SHAKE256::reset ( )
```

Reset state to feed and retrieve a new hash value.

squeeze_digest()

```
void Bse::SHAKE256::squeeze_digest (
    uint8_t * hashvalues,
    size_t n )
```

Retrieve an arbitrary number of hash value bytes.

update()

```
void Bse::SHAKE256::update (
    const uint8_t * data,
    size_t length )
```

Feed data to be hashed.

The documentation for this struct was generated from the following files:

- [bse/randomhash.hh](#)
- [bse/randomhash.cc](#)

2.108 Bse::SharedMemory Struct Reference

Descriptor for a shared memory region.

```
import "bseapi.idl";
```

Public Attributes

- [int64 shm_creator](#)
IPC id of the shared memory creator process.
- [int64 shm_start](#)
Shared memory area location.
- [int64 shm_length](#)
Shared memory area length in bytes.

Detailed Description

Descriptor for a shared memory region.

Member Data Documentation

shm_creator

`int64` Bse::SharedMemory::shm_creator

IPC id of the shared memory creator process.

shm_length

`int64` Bse::SharedMemory::shm_length

Shared memory area length in bytes.

shm_start

`int64` Bse::SharedMemory::shm_start

Shared memory area location.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.109 Bse::ShmFragment Struct Reference

Fragment description for interesting bits of shared memory.

```
import "bseapi.idl";
```

Detailed Description

Fragment description for interesting bits of shared memory.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.110 Bse::ShmFragmentSeq Struct Reference

Collection of shared memory fragments.

```
import "bseapi.idl";
```

Detailed Description

Collection of shared memory fragments.

The documentation for this struct was generated from the following file:

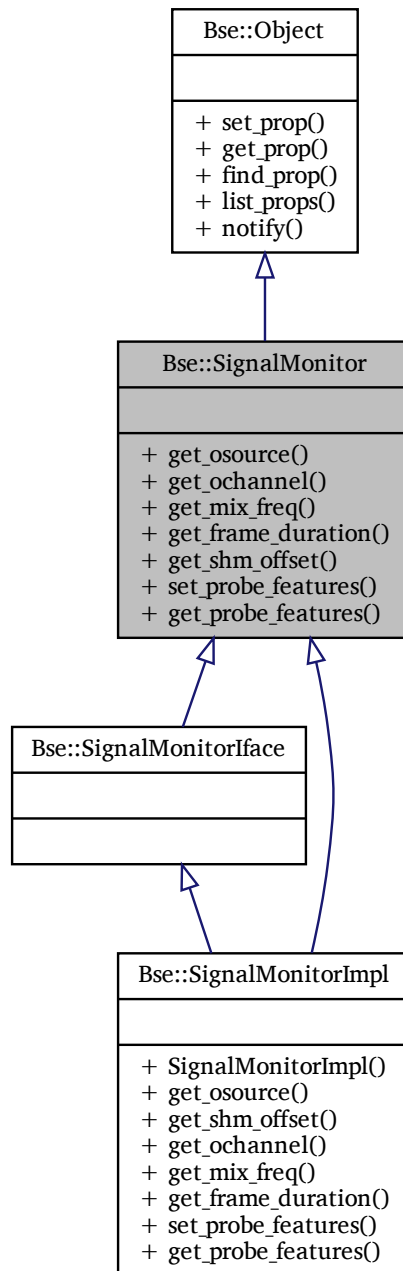
- [bse/bseapi.idl](#)

2.111 Bse::SignalMonitor Interface Reference

Interface for monitoring output signals.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::SignalMonitor:



Public Member Functions

- [Source get_osource \(\)](#)
Retrieve output module the *SignalMonitor* is connected to.
- [int32 get_ochannel \(\)](#)
Retrieve output channel the *SignalMonitor* is connected to.
- [int64 get_mix_freq \(\)](#)
Mix frequency at which monitor values are calculated.
- [int64 get_frame_duration \(\)](#)

Frame duration in μ seconds for the calculation of monitor values.

- `int64 get_shm_offset` (`MonitorField fld`)
Offset into shared memory for `MonitorField` values of `ochannel`.
- `void set_probe_features` (`ProbeFeatures pf`)
Configure probe features.
- `ProbeFeatures get_probe_features` ()
Get configured probe features.

Detailed Description

Interface for monitoring output signals.

Member Function Documentation

`get_frame_duration()`

`int64 Bse::SignalMonitor::get_frame_duration ()`
Frame duration in μ seconds for the calculation of monitor values.

`get_mix_freq()`

`int64 Bse::SignalMonitor::get_mix_freq ()`
Mix frequency at which monitor values are calculated.

`get_ochannel()`

`int32 Bse::SignalMonitor::get_ochannel ()`
Retrieve output channel the `SignalMonitor` is connected to.

`get_osource()`

`Source Bse::SignalMonitor::get_osource ()`
Retrieve output module the `SignalMonitor` is connected to.

`get_probe_features()`

`ProbeFeatures Bse::SignalMonitor::get_probe_features ()`
Get configured probe features.

`get_shm_offset()`

`int64 Bse::SignalMonitor::get_shm_offset (`
 `MonitorField fld)`
Offset into shared memory for `MonitorField` values of `ochannel`.

`set_probe_features()`

`void Bse::SignalMonitor::set_probe_features (`
 `ProbeFeatures pf)`
Configure probe features.
The documentation for this interface was generated from the following file:

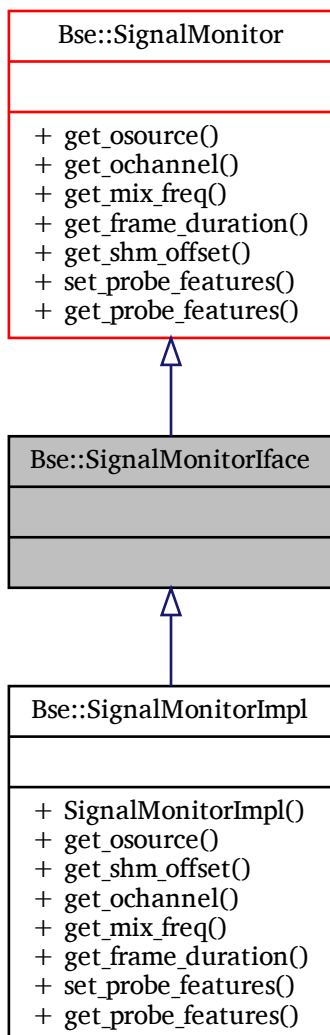
- `bse/bseapi.idl`

2.112 Bse::SignalMonitorIface Class Reference

IDL interface class for [Bse::SignalMonitor](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::SignalMonitorIface:



Additional Inherited Members

Detailed Description

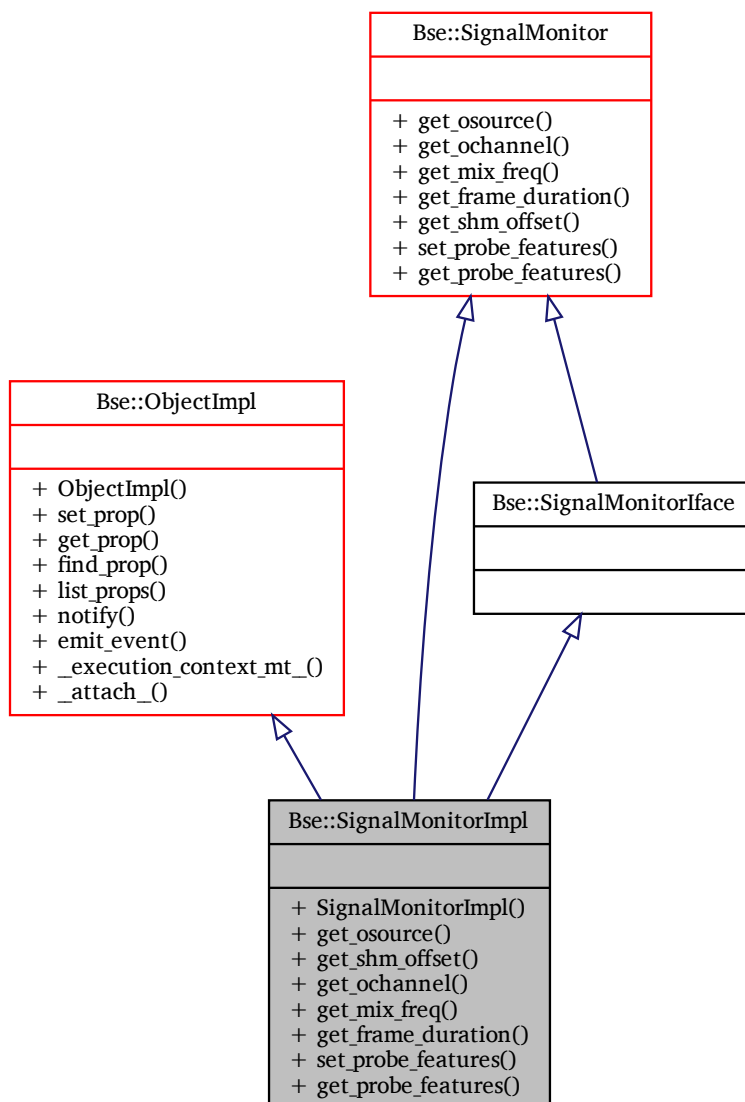
IDL interface class for [Bse::SignalMonitor](#).

The documentation for this class was generated from the following file:

- `bse/bseapi_interfaces.hh`

2.113 Bse::SignalMonitorImpl Class Reference

Inheritance diagram for Bse::SignalMonitorImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

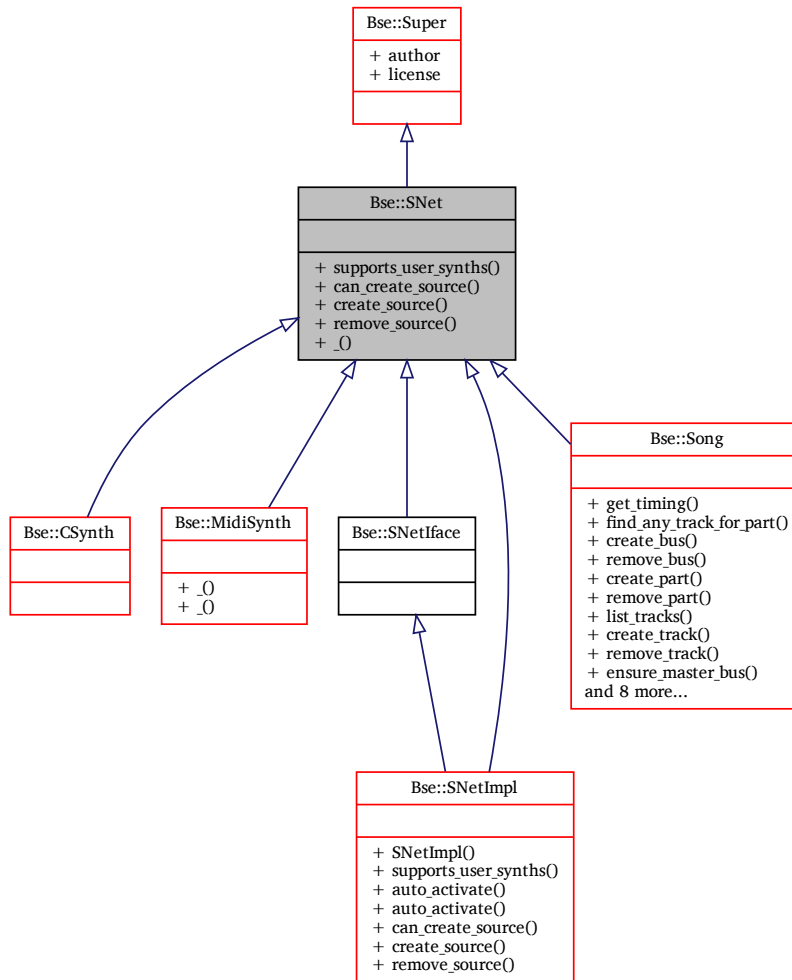
- `bse/monitor.hh`
- `bse/monitor.cc`

2.114 Bse::SNet Interface Reference

Base interface type for all kinds of synthesis networks.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::SNet:



Public Member Functions

- `bool supports_user_synths ()`
Check whether users may edit synthesis modules of this network.
- `Error can_create_source (String module_type)`
Check whether inserting a new module into a synthesis network is possible.
- `Source create_source (String module_type)`
Insert a new module into a synthesis network.
- `Error remove_source (Source module)`
Remove an existing module from its synthesis network.

Detailed Description

Base interface type for all kinds of synthesis networks.

Member Function Documentation

can_create_source()

Error Bse::SNet::can_create_source (
 [String](#) *module_type*)

Check whether inserting a new module into a synthesis network is possible.

create_source()

Source Bse::SNet::create_source (
 [String](#) *module_type*)

Insert a new module into a synthesis network.

remove_source()

Error Bse::SNet::remove_source (
 [Source](#) *module*)

Remove an existing module from its synthesis network.

supports_user_synths()

bool Bse::SNet::supports_user_synths ()

Check whether users may edit synthesis modules of this network.

The documentation for this interface was generated from the following file:

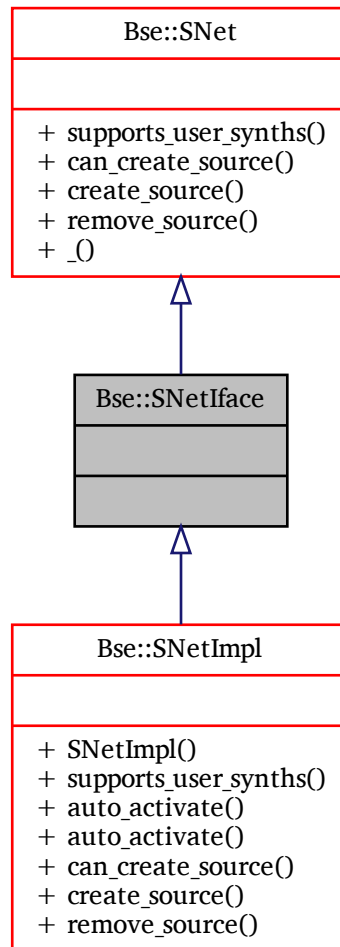
- [bse/bseapi.idl](#)

2.115 Bse::SNetIface Class Reference

IDL interface class for [Bse::SNet](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::SNetIface:



Additional Inherited Members

Detailed Description

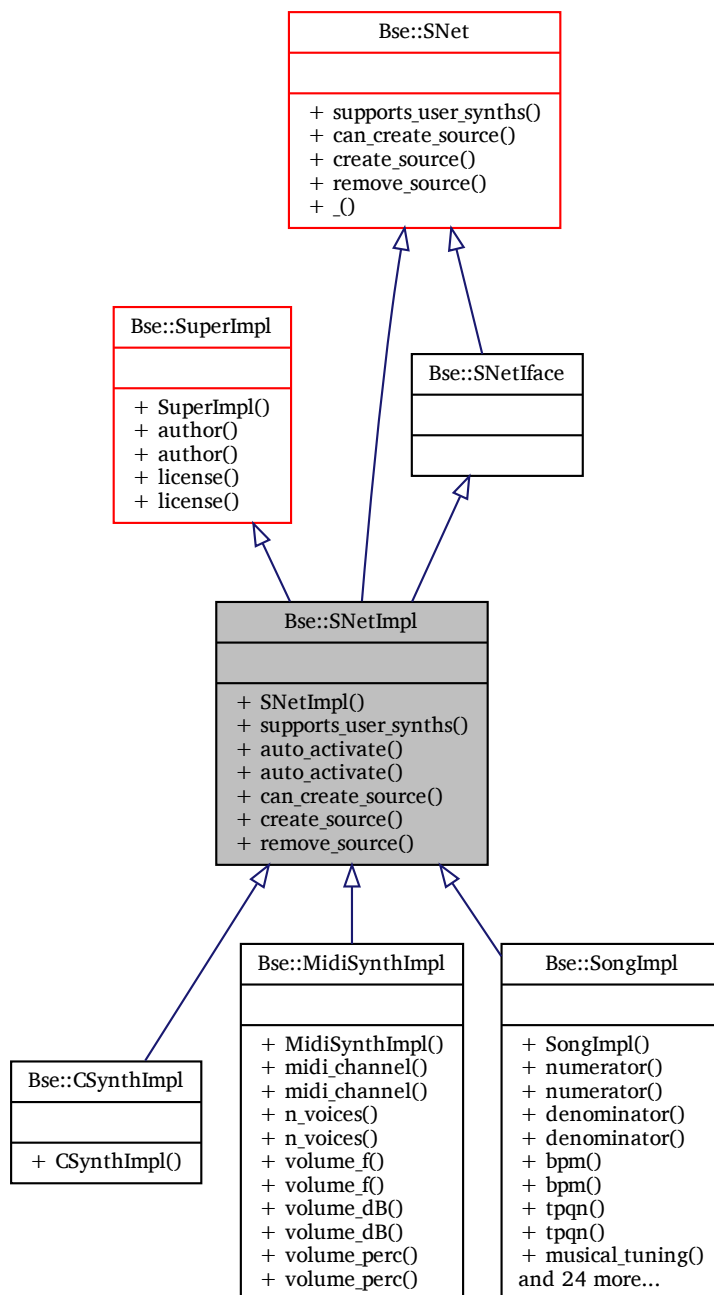
IDL interface class for [Bse::SNet](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.116 Bse::SNetImpl Class Reference

Inheritance diagram for Bse::SNetImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

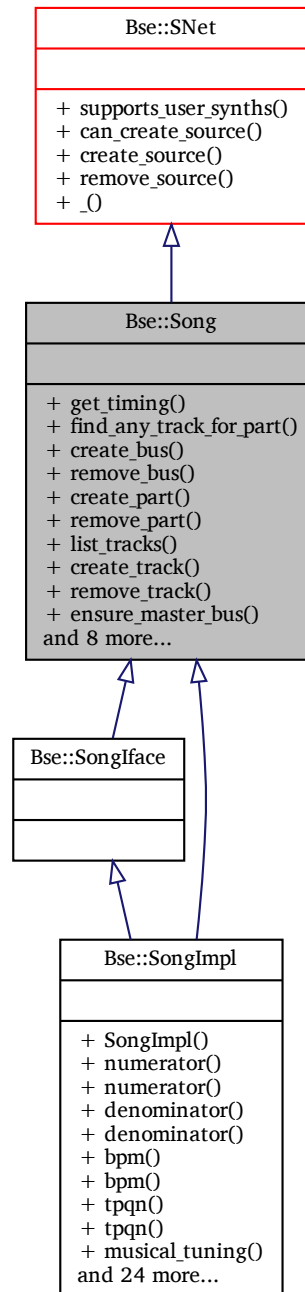
- bse/bsesnet.hh
- bse/bsesnet.cc

2.117 Bse::Song Interface Reference

Interface for [Track](#) and [Part](#) objects, as well as meta data for sequencing.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Song:



Public Member Functions

- [SongTiming](#) `get_timing` ([int32](#) tick)
Retrieve song timing information at a specific tick.
- [Track](#) `find_any_track_for_part` ([Part](#) part)

Find the first track that contains part, suitable to check for orphan parts.

- [Bus create_bus](#) ()
Create a new mixer bus for a [Song](#).
- [void remove_bus](#) ([Bus](#) bus)
Delete a mixer bus from a [Song](#).
- [Part create_part](#) ()
Create a new [Part](#) in a [Song](#).
- [void remove_part](#) ([Part](#) part)
Delete a [Part](#) from a [Song](#).
- [TrackSeq list_tracks](#) ()
List all tracks of this song.
- [Track create_track](#) ()
Create a new [Track](#) for a [Song](#).
- [void remove_track](#) ([Track](#) track)
Delete a [Track](#) from a [Song](#).
- [Bus ensure_master_bus](#) ()
Retrieve master output bus of a song, will create one if it doesn't exist.
- [void ensure_track_links](#) ()
Ensure that each part in a song is inserted into at least one track.
- [Track find_track_for_part](#) ([Part](#) part)
Find a track suitable for playing notes of a given part.
- [Bus get_master_bus](#) ()
Retrieve master output bus of a song if it exists.
- [void synthesize_note](#) ([Track](#) track, [int32](#) duration, [int32](#) note, [int32](#) fine_tune, [float64](#) velocity)
Synthesize a note on a song of an active project.
- [int64 get_shm_offset](#) ([SongTelemetry](#) fld)
Offset into [SharedMemory](#) for [SongTelemetry](#) fields.

Detailed Description

Interface for [Track](#) and [Part](#) objects, as well as meta data for sequencing.

Member Function Documentation

[create_bus](#)()

[Bus](#) [Bse::Song::create_bus](#) ()
Create a new mixer bus for a [Song](#).

[create_part](#)()

[Part](#) [Bse::Song::create_part](#) ()
Create a new [Part](#) in a [Song](#).

[create_track](#)()

[Track](#) [Bse::Song::create_track](#) ()
Create a new [Track](#) for a [Song](#).

ensure_master_bus()

`Bus Bse::Song::ensure_master_bus ()`

Retrieve master output bus of a song, will create one if it doesn't exist.

ensure_track_links()

`void Bse::Song::ensure_track_links ()`

Ensure that each part in a song is inserted into at least one track.

find_any_track_for_part()

`Track Bse::Song::find_any_track_for_part (`
`Part part)`

Find the first track that contains part, suitable to check for orphan parts.

find_track_for_part()

`Track Bse::Song::find_track_for_part (`
`Part part)`

Find a track suitable for playing notes of a given part.

get_master_bus()

`Bus Bse::Song::get_master_bus ()`

Retrieve master output bus of a song if it exists.

get_shm_offset()

`int64 Bse::Song::get_shm_offset (`
`SongTelemetry fld)`

Offset into `SharedMemory` for `SongTelemetry` fields.

get_timing()

`SongTiming Bse::Song::get_timing (`
`int32 tick)`

Retrieve song timing information at a specific tick.

list_tracks()

`TrackSeq Bse::Song::list_tracks ()`

List all tracks of this song.

remove_bus()

`void Bse::Song::remove_bus (`
`Bus bus)`

Delete a mixer bus from a `Song`.

remove_part()

```
void Bse::Song::remove_part (
    Part part )
```

Delete a [Part](#) from a [Song](#).

remove_track()

```
void Bse::Song::remove_track (
    Track track )
```

Delete a [Track](#) from a [Song](#).

synthesize_note()

```
void Bse::Song::synthesize_note (
    Track track,
    int32 duration,
    int32 note,
    int32 fine_tune,
    float64 velocity )
```

Synthesize a note on a song of an active project.

The documentation for this interface was generated from the following file:

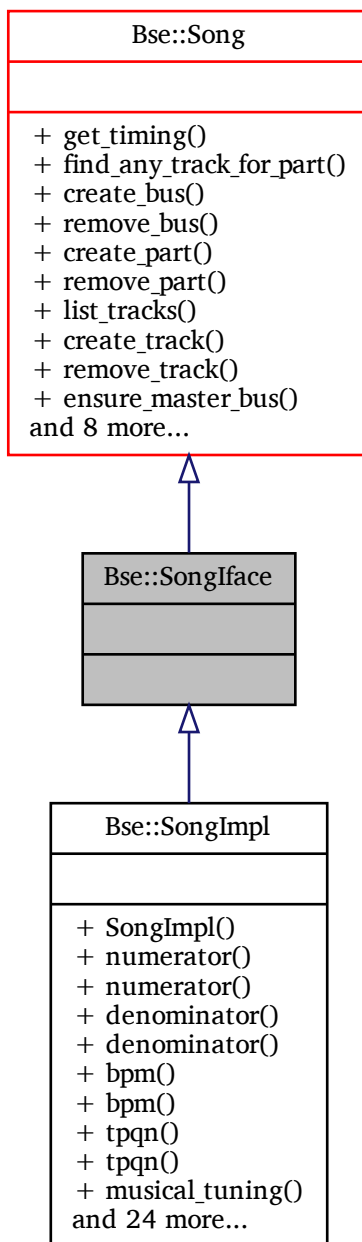
- [bse/bseapi.idl](#)

2.118 Bse::SongIface Class Reference

IDL interface class for [Bse::Song](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::SongIface:



Additional Inherited Members

Detailed Description

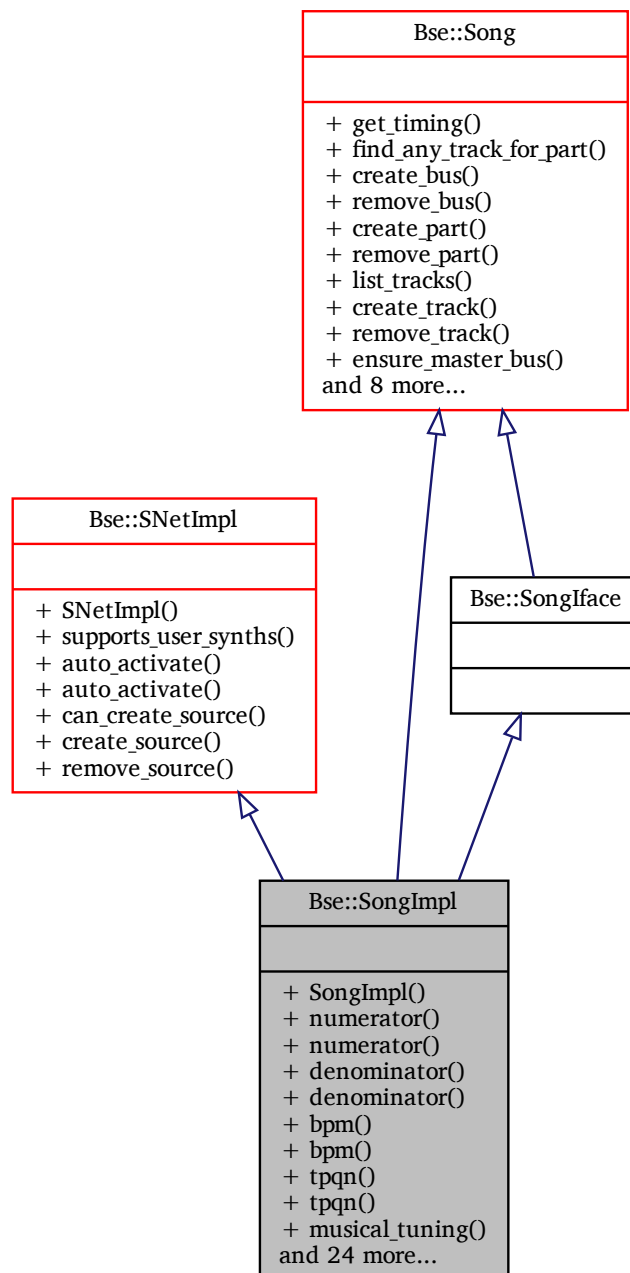
IDL interface class for [Bse::Song](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.119 Bse::SongImpl Class Reference

Inheritance diagram for Bse::SongImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- `bse/bsesong.hh`
- `bse/bsesong.cc`

2.120 Bse::SongTiming Struct Reference

Song timing configuration.

```
import "bseapi.idl";
```

Detailed Description

Song timing configuration.

The documentation for this struct was generated from the following file:

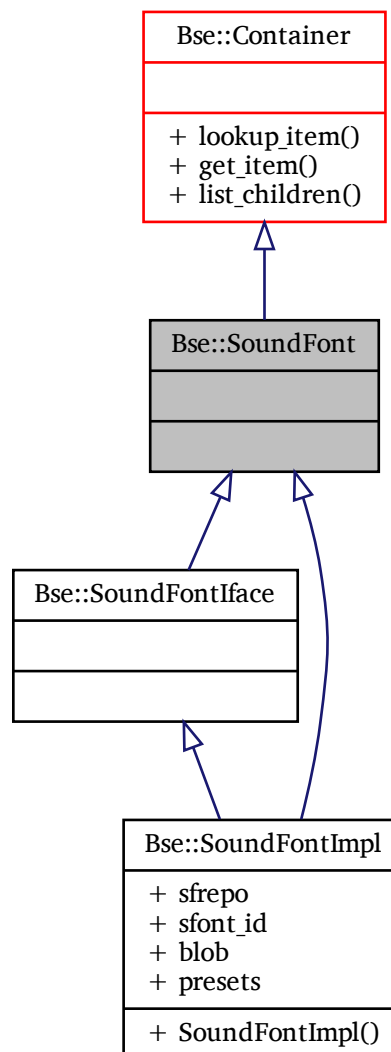
- [bse/bseapi.idl](#)

2.121 Bse::SoundFont Interface Reference

Interface for sound fonts.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::SoundFont:



Additional Inherited Members

Detailed Description

Interface for sound fonts.

The documentation for this interface was generated from the following file:

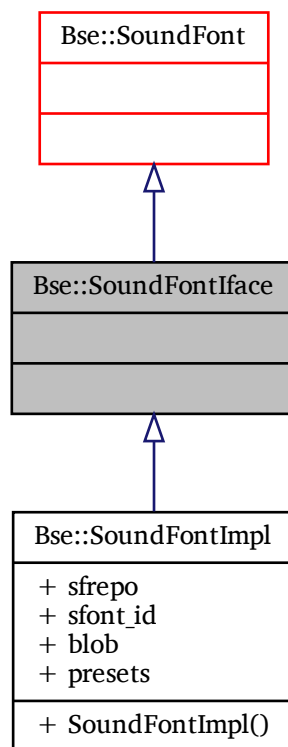
- [bse/bseapi.idl](#)

2.122 Bse::SoundFontIface Class Reference

IDL interface class for [Bse::SoundFont](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::SoundFontIface:



Additional Inherited Members

Detailed Description

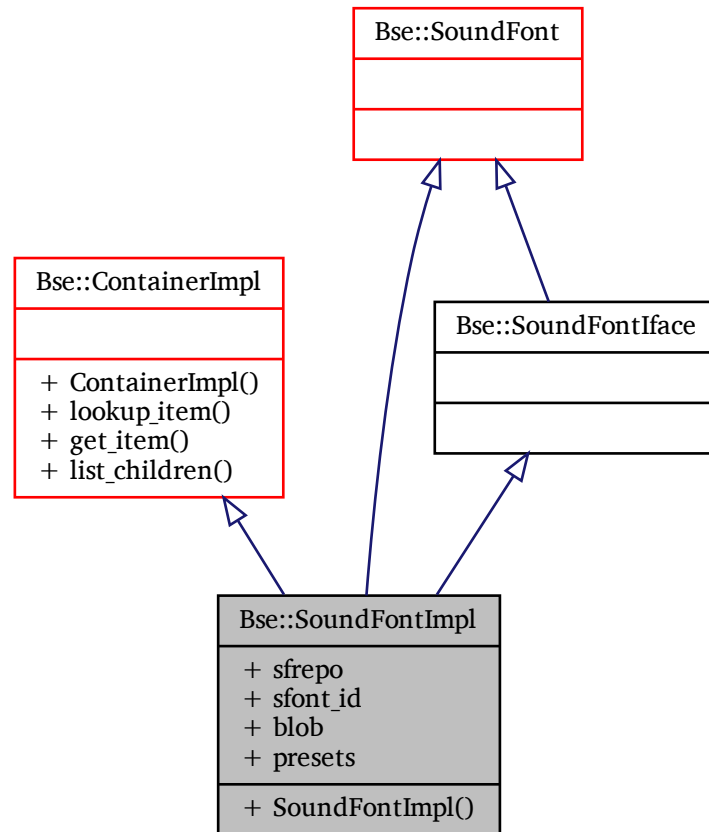
IDL interface class for [Bse::SoundFont](#).

The documentation for this class was generated from the following file:

- [bse/bseapi_interfaces.hh](#)

2.123 Bse::SoundFontImpl Class Reference

Inheritance diagram for Bse::SoundFontImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

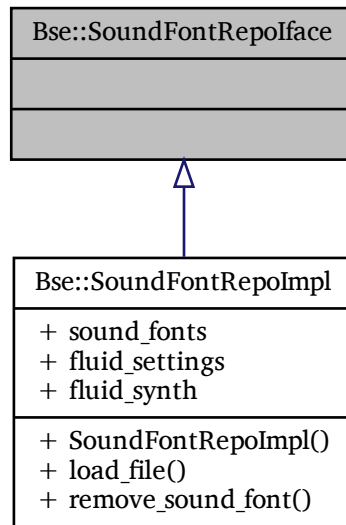
- bse/bsesoundfont.hh
- bse/bsesoundfont.cc

2.124 Bse::SoundFontRepoIface Class Reference

IDL interface class for Bse::SoundFontRepo.

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::SoundFontRepoiface:



Detailed Description

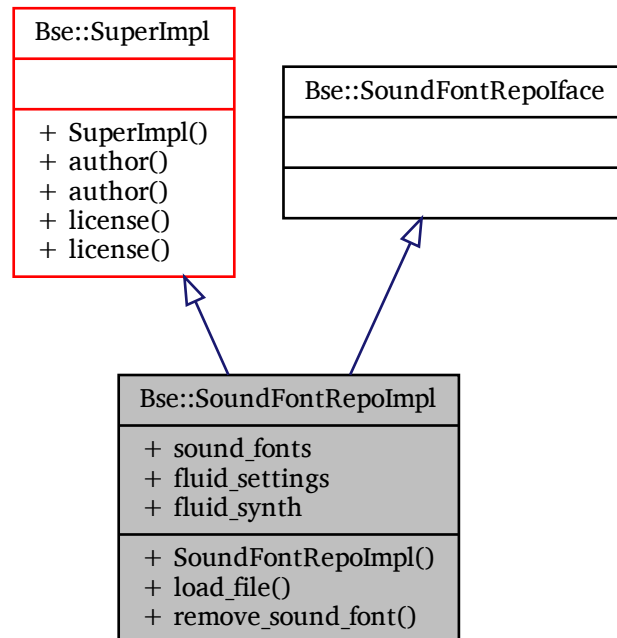
IDL interface class for Bse::SoundFontRepo.

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.125 Bse::SoundFontRepoImpl Class Reference

Inheritance diagram for Bse::SoundFontRepoImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

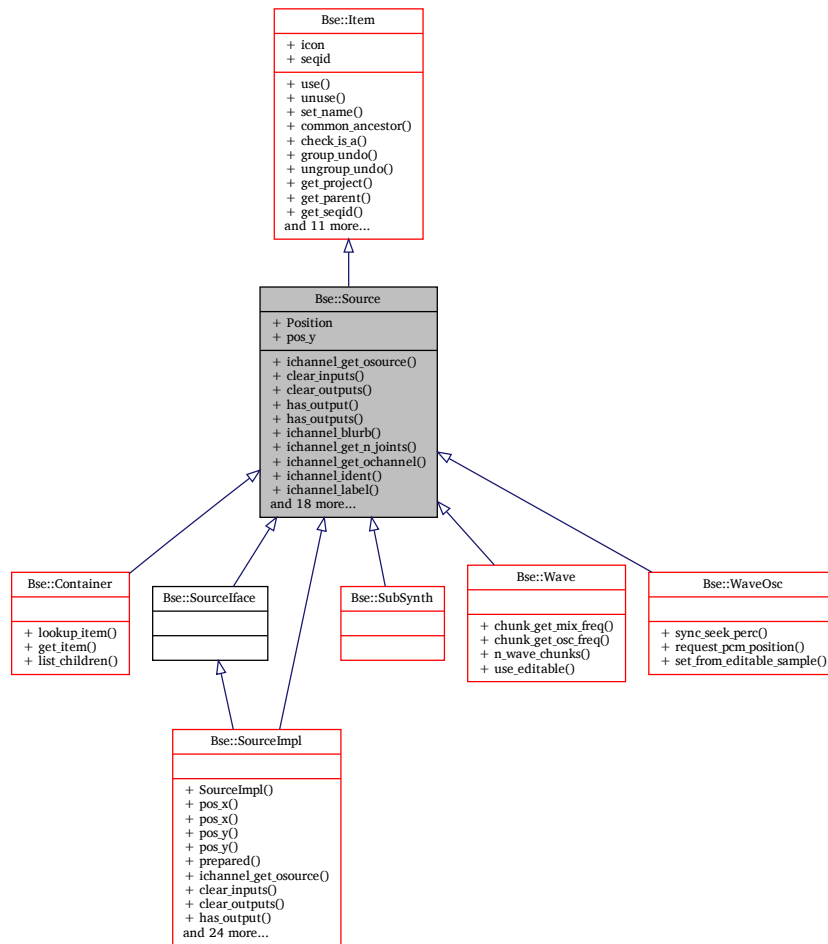
- bse/bsesoundfontrepo.hh
- bse/bsesoundfontrepo.cc

2.126 Bse::Source Interface Reference

Base interface type for synthesis modules with input or output streams.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Source:



Public Member Functions

- **Source** `ichannel_get_osource (int32 input_channel, int32 input_joint)`
Retrieve output module connected to a specific joint of an input channel.
- **void** `clear_inputs ()`
Disconnect all module inputs.
- **void** `clear_outputs ()`
Disconnect all module outputs.
- **bool** `has_output (int32 ochannel)`
Check whether a module's output channel is connected.
- **bool** `has_outputs ()`
Check whether a module has output channel connections.
- **String** `ichannel_blurb (int32 input_channel)`
Get input channel description.
- **int32** `ichannel_get_n_joints (int32 input_channel)`
Retrieve the number of inputs connected to an input channel.
- **int32** `ichannel_get_ochannel (int32 input_channel, int32 input_joint)`
Retrieve output channel of the module connected to a specific joint of an input channel.
- **String** `ichannel_ident (int32 input_channel)`
Get canonical input channel name.

- **String** `ichannel_label` (`int32` `input_channel`)
Get input channel name.
- **bool** `is_joint_ichannel` (`String` `input_channel`)
Check if an input channel is a joint (multi-connect) channel.
- **bool** `is_joint_ichannel_by_id` (`int32` `input_channel`)
Check if an input channel is a joint (multi-connect) channel.
- **bool** `is_prepared` ()
Check whether a source is prepared for synthesis processing.
- `int32` `n_ichannels` ()
Get the number of input channels of a module.
- `int32` `n_ochannels` ()
Get the number of output channels of a module.
- **String** `ochannel_blurb` (`int32` `output_channel`)
Get output channel description.
- **String** `ochannel_ident` (`int32` `output_channel`)
Get canonical output channel name.
- **String** `ochannel_label` (`int32` `output_channel`)
Get output channel name.
- **Error** `set_automation` (`String` `property_name`, `int32` `midi_channel`, `MidiControl` `control_type`)
Setup automation parameters for a property.
- `MidiControl` `get_automation_control` (`String` `property_name`)
Get control type from an automation property.
- `int32` `get_automation_channel` (`String` `property_name`)
Get MIDI channel from an automation property.
- **Error** `set_input` (`String` `input_channel`, `Source` `omodule`, `String` `output_channel`)
Connect a module input to another module's output.
- **Error** `set_input_by_id` (`int32` `input_channel`, `Source` `omodule`, `int32` `output_channel`)
Connect a module input to another module's output.
- **Error** `unset_input` (`String` `input_channel`, `Source` `omodule`, `String` `output_channel`)
Disconnect a module input.
- **Error** `unset_input_by_id` (`int32` `input_channel`, `Source` `omodule`, `int32` `output_channel`)
Disconnect a module input.
- **void** `set_pos` (`float64` `x_pos`, `float64` `y_pos`)
Set the x and y position of a module.
- `int32` `get_mix_freq` ()
Retrieve the current mixing frequency used for probes.
- **SignalMonitor** `create_signal_monitor` (`int32` `ochannel`)
Create signal monitor for an output channel.

Detailed Description

Base interface type for synthesis modules with input or output streams.

Member Function Documentation

`clear_inputs()`

```
void Bse::Source::clear_inputs ( )
```

Disconnect all module inputs.

clear_outputs()

```
void Bse::Source::clear_outputs ( )
```

Disconnect all module outputs.

create_signal_monitor()

```
SignalMonitor Bse::Source::create_signal_monitor (
    int32 ochannel )
```

Create signal monitor for an output channel.

get_automation_channel()

```
int32 Bse::Source::get_automation_channel (
    String property_name )
```

Get MIDI channel from an automation property.

get_automation_control()

```
MidiControl Bse::Source::get_automation_control (
    String property_name )
```

Get control type from an automation property.

get_mix_freq()

```
int32 Bse::Source::get_mix_freq ( )
```

Retrieve the current mixing frequency used for probes.

has_output()

```
bool Bse::Source::has_output (
    int32 ochannel )
```

Check whether a module's output channel is connected.

has_outputs()

```
bool Bse::Source::has_outputs ( )
```

Check whether a module has output channel connections.

ichannel_blurb()

```
String Bse::Source::ichannel_blurb (
    int32 input_channel )
```

Get input channel description.

ichannel_get_n_joints()

```
int32 Bse::Source::ichannel_get_n_joints (
    int32 input_channel )
```

Retrieve the number of inputs connected to an input channel.

ichannel_get_ochannel()

```
int32 Bse::Source::ichannel_get_ochannel (
    int32 input_channel,
    int32 input_joint )
```

Retrieve output channel of the module connected to a specific joint of an input channel.

ichannel_get_osource()

```
Source Bse::Source::ichannel_get_osource (
    int32 input_channel,
    int32 input_joint )
```

Retrieve output module connected to a specific joint of an input channel.

ichannel_ident()

```
String Bse::Source::ichannel_ident (
    int32 input_channel )
```

Get canonical input channel name.

ichannel_label()

```
String Bse::Source::ichannel_label (
    int32 input_channel )
```

Get input channel name.

is_joint_ichannel()

```
bool Bse::Source::is_joint_ichannel (
    String input_channel )
```

Check if an input channel is a joint (multi-connect) channel.

is_joint_ichannel_by_id()

```
bool Bse::Source::is_joint_ichannel_by_id (
    int32 input_channel )
```

Check if an input channel is a joint (multi-connect) channel.

is_prepared()

```
bool Bse::Source::is_prepared ( )
```

Check whether a source is prepared for synthesis processing.

n_ichannels()

```
int32 Bse::Source::n_ichannels ( )
```

Get the number of input channels of a module.

n_ochannels()

```
int32 Bse::Source::n_ochannels ( )
```

Get the number of output channels of a module.

ochannel_blurb()

```
String Bse::Source::ochannel_blurb (
    int32 output_channel )
```

Get output channel description.

ochannel_ident()

```
String Bse::Source::ochannel_ident (
    int32 output_channel )
```

Get canonical output channel name.

ochannel_label()

```
String Bse::Source::ochannel_label (
    int32 output_channel )
```

Get output channel name.

set_automation()

```
Error Bse::Source::set_automation (
    String property_name,
    int32 midi_channel,
    MidiControl control_type )
```

Setup automation parameters for a property.

Parameters

<i>midi_channel</i>	The MIDI Channel from which automation events should be received, 0 designates the default MIDI channel
<i>control_type</i>	The type of control events used for automation

set_input()

```
Error Bse::Source::set_input (
    String input_channel,
    Source omodule,
    String output_channel )
```

Connect a module input to another module's output.

set_input_by_id()

```
Error Bse::Source::set_input_by_id (
    int32 input_channel,
    Source omodule,
    int32 output_channel )
```

Connect a module input to another module's output.

set_pos()

```
void Bse::Source::set_pos (
    float64 x_pos,
    float64 y_pos )
```

Set the x and y position of a module.

In contrast to setting the position through ordinary object property setters, this function will not update the module position if the passed in arguments are sufficiently equal to the values already set on the object. As such, it does not record an extra undo step for setting properties to values they already have and if necessary turns setting of x and y positions into an atomic undo operation.

unset_input()

```
Error Bse::Source::unset_input (
    String input_channel,
    Source omodule,
    String output_channel )
```

Disconnect a module input.

unset_input_by_id()

```
Error Bse::Source::unset_input_by_id (
    int32 input_channel,
    Source omodule,
    int32 output_channel )
```

Disconnect a module input.

The documentation for this interface was generated from the following file:

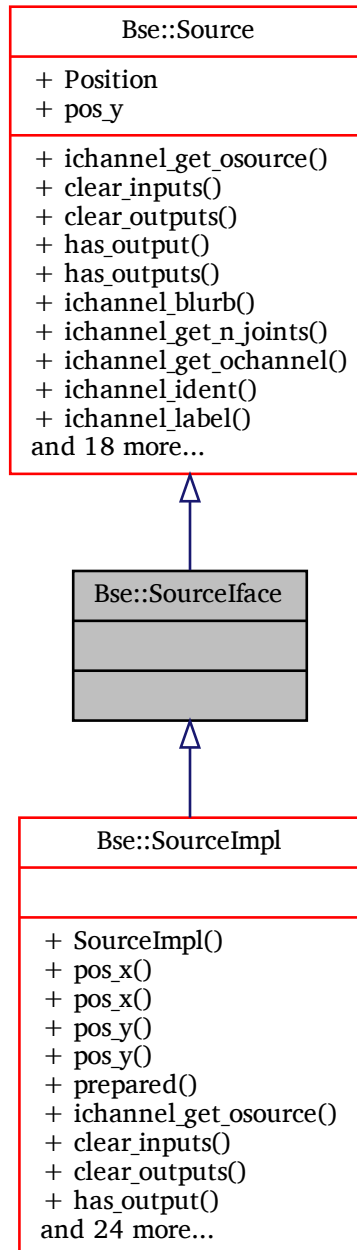
- [bse/bseapi.idl](#)

2.127 Bse::Sourceface Class Reference

IDL interface class for [Bse::Source](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::Sourceface:



Additional Inherited Members

Detailed Description

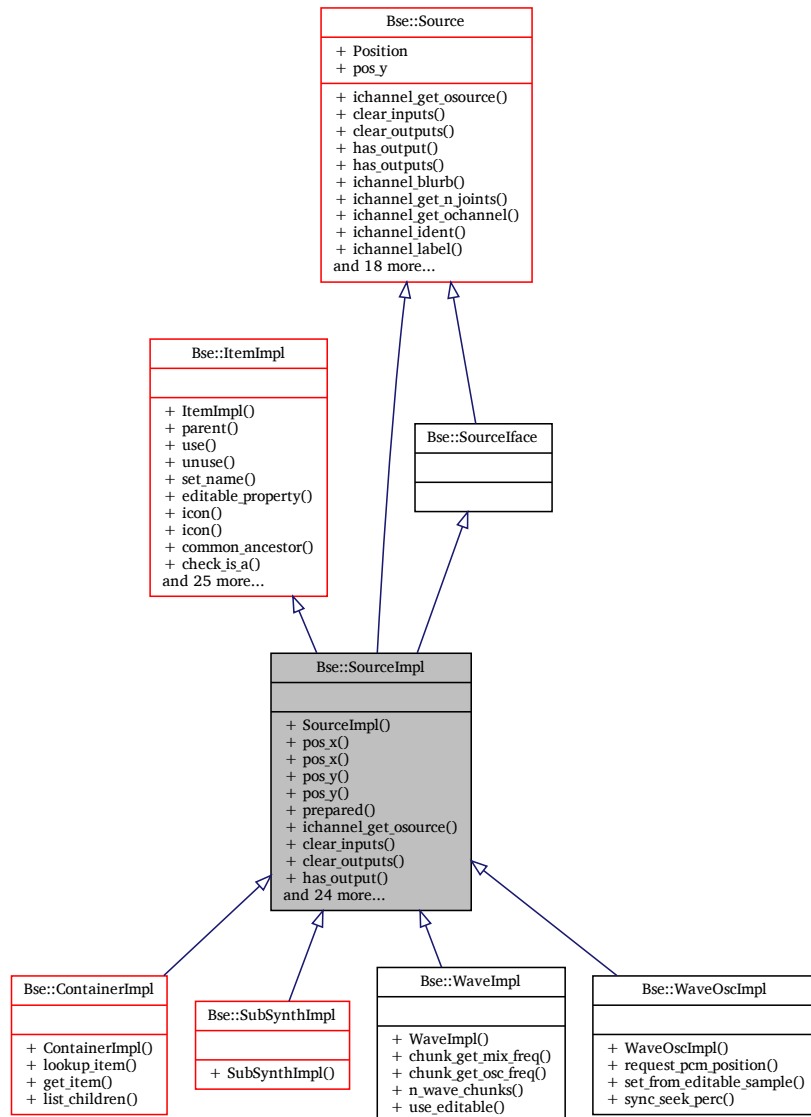
IDL interface class for [Bse::Source](#).

The documentation for this class was generated from the following file:

- [bse/bseapi_interfaces.hh](#)

2.128 Bse::SourceImpl Class Reference

Inheritance diagram for Bse::SourceImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- bse/bsesource.hh
- bse/bsesource.cc
- bse/monitor.cc

2.129 Bse::Spinlock Class Reference

The **Spinlock** uses low-latency busy spinning to acquire locks.

```
#include <bcore.hh>
```

Detailed Description

The [Spinlock](#) uses low-latency busy spinning to acquire locks.

This class is a thin wrapper around `pthread_spin_lock()` and related functions. This class supports static construction.

The documentation for this class was generated from the following file:

- bse/bcore.hh

2.130 Bse::Lib::StringFormatter Class Reference

[StringFormatter](#) - `sprintf()` like string formatting for C++.

```
#include <formatter.hh>
```

Static Public Member Functions

- `template<LocaleContext LC = POSIX_LOCALE, class ... Args>`
`static std::string format (const ArgTransform &arg_transform, const char *format, const Args &...arguments)`

Format a string according to an `sprintf()` format string with arguments.

Detailed Description

[StringFormatter](#) - `sprintf()` like string formatting for C++.

See `format()` for supported flags, modifiers and conversions. To find source code strings with size modifiers for possible cleanups, use: `egrep "\\"([^\"]|\\")*%[0-9$]*[-+#0\I]*[0-9$][.]*[0-9$]*[hLlqjzt] + [nSspmCdiouXx←FfGgEeAa]"`

Member Function Documentation

format()

```
template<LocaleContext LC = POSIX_LOCALE, class ... Args>
static std::string Bse::Lib::StringFormatter::format (
    const ArgTransform & arg_transform,
    const char * format,
    const Args &... arguments ) [inline], [static]
```

Format a string according to an `sprintf()` *format* string with *arguments*.

Refer to `sprintf()` for the format string details, this function is designed to serve as an `sprintf()` replacement and mimic its behaviour as close as possible. Supported format directive features are:

- Formatting flags (sign conversion, padding, alignment), i.e. the flags: `[-#0+ ']`
- Field width and precision specifications.
- Positional arguments for field width, precision and value.
- Length modifiers are tolerated: i.e. any of `[hLljqzZ]`.
- The conversion specifiers `[spmcCdiouXxFfGgEeAa]`.

Additionally, arguments can be transformed after conversion by passing a `std::string` conversion function as *arg transform*. This may e.g. be used for XML character escaping of the format argument values.

Note: Format errors, e.g. missing arguments will produce a warning on `stderr` and return the *format* string unmodified.

Returns

A formatted string.

The documentation for this class was generated from the following files:

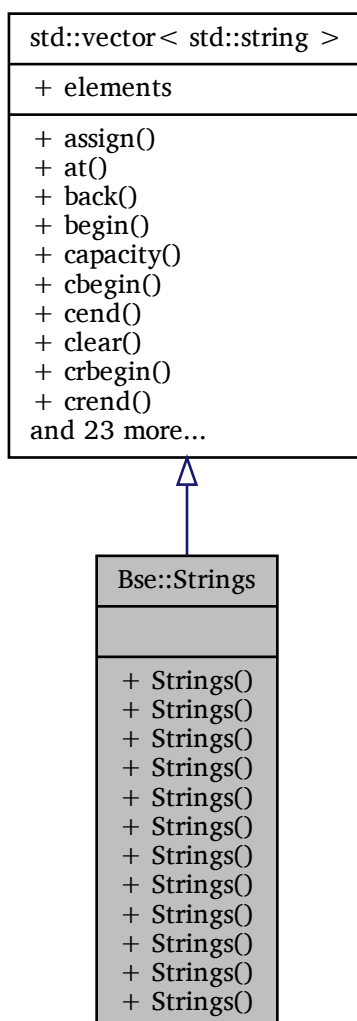
- bse/formatter.hh
- bse/formatter.cc

2.131 Bse::Strings Class Reference

Convenience Constructor for [StringSeq](#) or `std::vector<std::string>`

```
#include <strings.hh>
```

Inheritance diagram for Bse::Strings:



Additional Inherited Members

Detailed Description

Convenience Constructor for [StringSeq](#) or `std::vector<std::string>`

The documentation for this class was generated from the following files:

- [bse/strings.hh](#)
- [bse/strings.cc](#)

2.132 Bse::StringSeq Struct Reference

Stringeq - a variable length list of test strings.

```
import "bseapi.idl";
```

Detailed Description

Stringeq - a variable length list of test strings.

The documentation for this struct was generated from the following file:

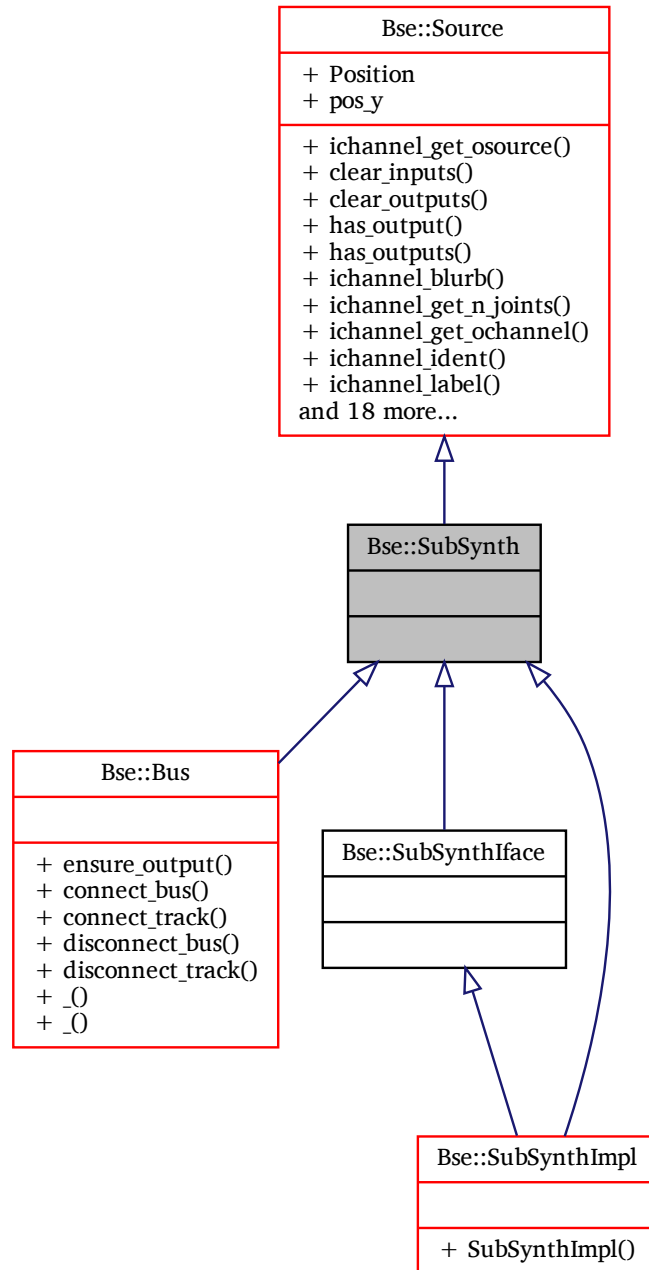
- [bse/bseapi.idl](#)

2.133 Bse::SubSynth Interface Reference

Synthesizer module for embedding (rerouting input and output) of another synthesizer network ([SNet](#)).

```
import "bseapi.idl";
```

Inheritance diagram for Bse::SubSynth:



Additional Inherited Members

Detailed Description

Synthesizer module for embedding (rerouting input and output) of another synthesizer network (*SNet*). The documentation for this interface was generated from the following file:

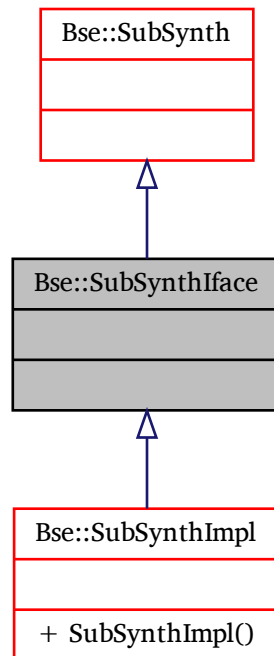
- [bse/bseapi.idl](#)

2.134 Bse::SubSynthIface Class Reference

IDL interface class for [Bse::SubSynth](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::SubSynthIface:



Additional Inherited Members

Detailed Description

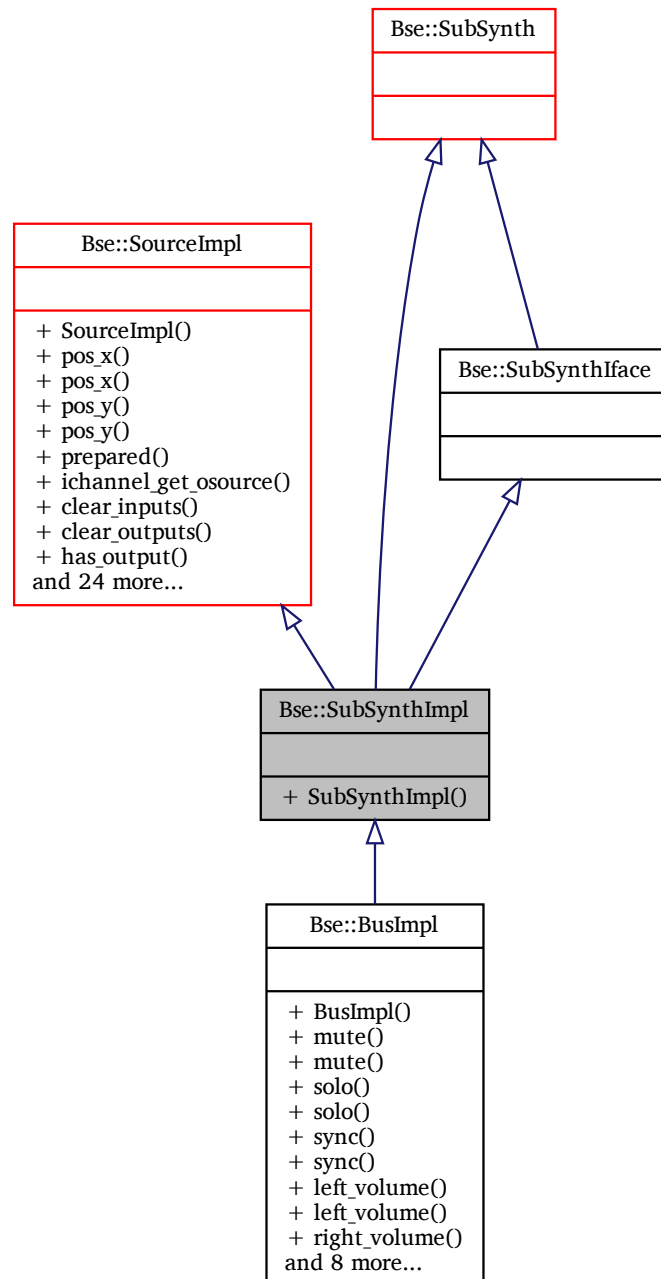
IDL interface class for [Bse::SubSynth](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.135 Bse::SubSynthImpl Class Reference

Inheritance diagram for Bse::SubSynthImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

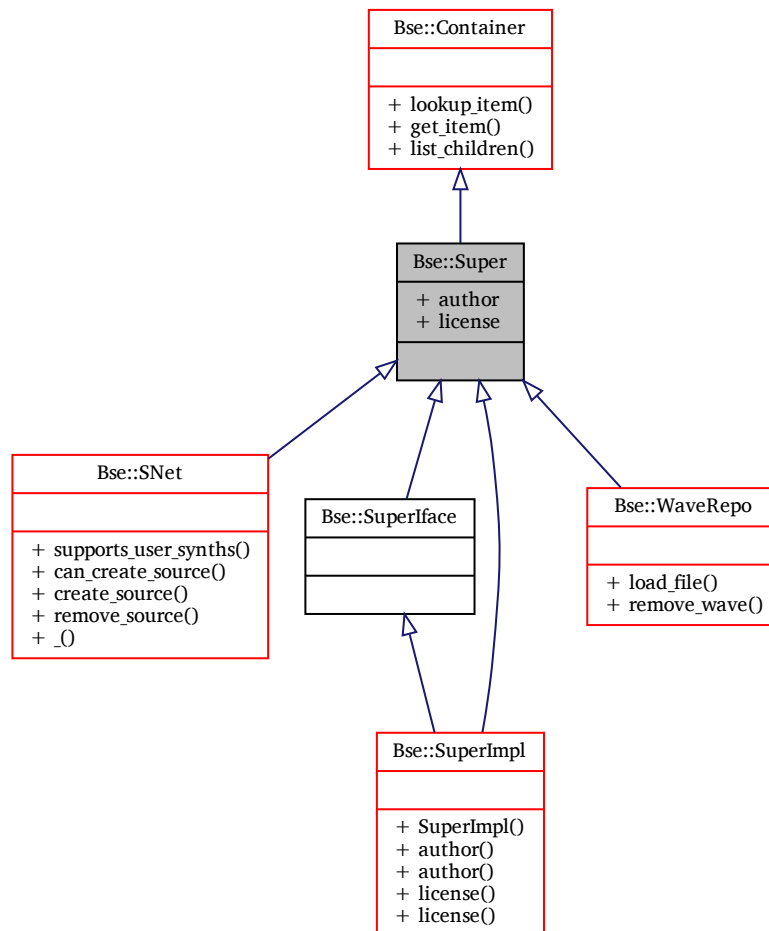
- `bse/bsesubsynth.hh`
- `bse/bsesubsynth.cc`

2.136 Bse::Super Interface Reference

Base interface type for [Item](#) managers.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Super:



Additional Inherited Members

Detailed Description

Base interface type for [Item](#) managers.

The documentation for this interface was generated from the following file:

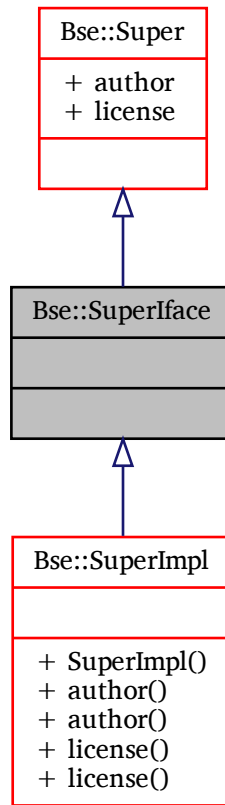
- [bse/bseapi.idl](#)

2.137 Bse::SuperInterface Class Reference

IDL interface class for [Bse::Super](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::SuperInterface:



Additional Inherited Members

Detailed Description

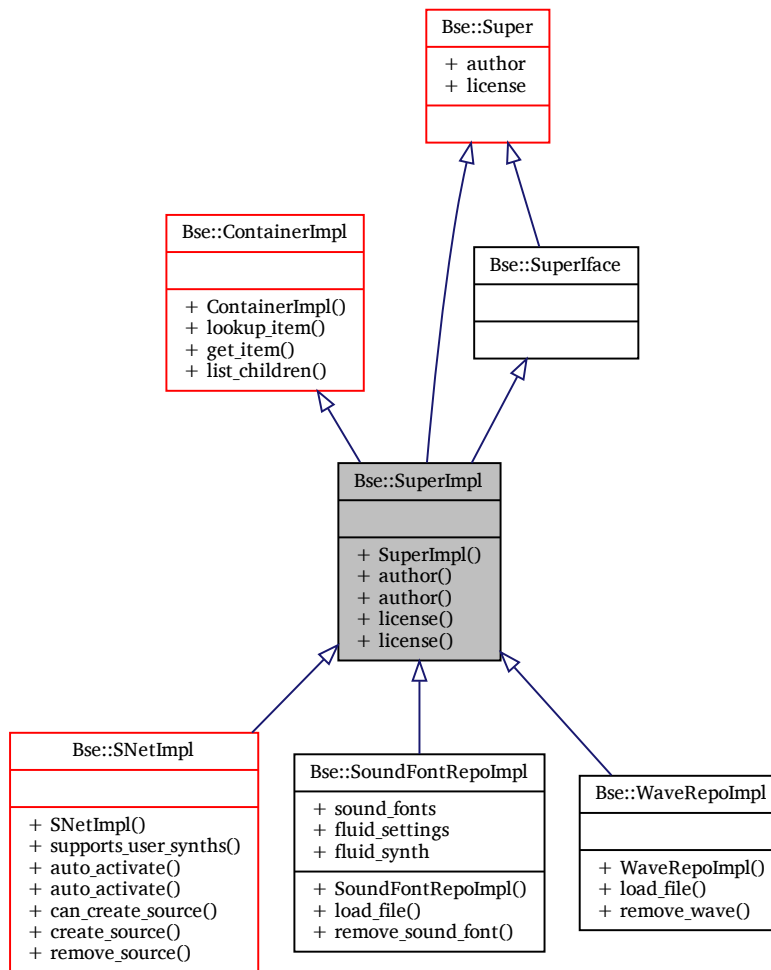
IDL interface class for [Bse::Super](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.138 Bse::SuperImpl Class Reference

Inheritance diagram for Bse::SuperImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- bse/bsesuper.hh
- bse/bsesuper.cc

2.139 Bse::SuperSeq Struct Reference

A list of [Super](#) type objects.

```
import "bseapi.idl";
```

Detailed Description

A list of [Super](#) type objects.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.140 Bse::TaskRegistry Class Reference

The task registry keeps track of runtime threads for profiling and statistical purposes.

```
#include <bsestartup.hh>
```

Static Public Member Functions

- static void **add** (const **std::string** &name, **int** pid, **int** tid = -1)
Add process/thread to registry for runtime profiling.
- static bool **remove** (**int** tid)
Remove process/thread based on thread_id.
- static void **update** ()
Issue [TaskStatus.update](#) on all tasks in registry.
- static **List** **list** ()
Retrieve a copy to the list of all tasks in registry.

Detailed Description

The task registry keeps track of runtime threads for profiling and statistical purposes.

Member Function Documentation

add()

```
void Bse::TaskRegistry::add (
    const std::string & name,
    int pid,
    int tid = -1 ) [static]
```

Add process/thread to registry for runtime profiling.

list()

```
TaskRegistry::List Bse::TaskRegistry::list ( ) [static]
```

Retrieve a copy to the list of all tasks in registry.

remove()

```
bool Bse::TaskRegistry::remove (
    int tid ) [static]
```

Remove process/thread based on thread_id.

update()

```
void Bse::TaskRegistry::update ( ) [static]
```

Issue [TaskStatus.update](#) on all tasks in registry.

The documentation for this class was generated from the following files:

- [bse/bsestartup.hh](#)
- [bse/bsestartup.cc](#)

2.141 Bse::TaskStatus Struct Reference

Acquire information about a task (process or thread) at runtime.

```
#include <platform.hh>
```

Public Member Functions

- [TaskStatus](#) (int pid, int tid = -1)
Construct from process ID and optionally thread ID.
- bool [update](#) ()
Update status information, might return false if called too frequently.
- [String](#) [string](#) ()
Retrieve string representation of the status information.

Public Attributes

- int [process_id](#)
Process ID.
- int [task_id](#)
Process ID or thread ID.
- [String](#) [name](#)
Thread name (set by user).
- State [state](#)
Thread state.
- int [processor](#)
Running processor number.
- int [priority](#)
Priority or nice value.
- uint64 [utime](#)
Userspace time.
- uint64 [stime](#)
System time.
- uint64 [cutime](#)
Userspace time of dead children.
- uint64 [cstime](#)
System time of dead children.
- uint64 [ac_stamp](#)
Accounting stamp.

Detailed Description

Acquire information about a task (process or thread) at runtime.

Constructor & Destructor Documentation

TaskStatus()

```
Bse::TaskStatus::TaskStatus (
    int pid,
    int tid = -1 ) [explicit]
```

Construct from process ID and optionally thread ID.

Member Function Documentation

string()

`String Bse::TaskStatus::string ()`

Retrieve string representation of the status information.

update()

`bool Bse::TaskStatus::update ()`

Update status information, might return false if called too frequently.

Member Data Documentation

ac_stamp

`uint64 Bse::TaskStatus::ac_stamp`

Accounting stamp.

cstime

`uint64 Bse::TaskStatus::cstime`

System time of dead children.

cutime

`uint64 Bse::TaskStatus::cutime`

Userspace time of dead children.

name

`String Bse::TaskStatus::name`

Thread name (set by user).

priority

`int Bse::TaskStatus::priority`

Priority or nice value.

process_id

`int Bse::TaskStatus::process_id`

Process ID.

processor

`int Bse::TaskStatus::processor`

Running processor number.

state

State `Bse::TaskStatus::state`
Thread state.

stime

`uint64 Bse::TaskStatus::stime`
System time.

task_id

`int Bse::TaskStatus::task_id`
Process ID or thread ID.

utime

`uint64 Bse::TaskStatus::utime`
Userspace time.

The documentation for this struct was generated from the following files:

- `bse/platform.hh`
- `bse/platform.cc`

2.142 Bse::Test::Timer Class Reference

Class for profiling benchmark tests.

```
#include <testing.hh>
```

Public Member Functions

- **Timer** (`double` `deadline_in_secs` = 0)
Create a `Timer()` instance, specifying an optional upper bound for test durations.
- `int64 n_reps` () const
Number of benchmark repetitions to execute.
- `double test_elapsed` () const
Seconds spent in `benchmark()`
- `double min_elapsed` () const
Minimum time benchmarked for a `callee()` call.
- `double max_elapsed` () const
Maximum time benchmarked for a `callee()` call.
- `template<typename Callee >`
`double benchmark` (`Callee` `callee`)

Detailed Description

Class for profiling benchmark tests.

UseCase: Benchmarking function implementations, e.g. to compare sorting implementations.

Constructor & Destructor Documentation

Timer()

```
Bse::Test::Timer::Timer (
    double deadline_in_secs = 0 ) [explicit]
```

Create a [Timer\(\)](#) instance, specifying an optional upper bound for test durations.

Member Function Documentation

benchmark()

```
template<typename Callee >
double Bse::Test::Timer::benchmark (
    Callee callee )
```

Parameters

<i>callee</i>	A callable function or object. Method to benchmark the execution time of <i>callee</i> .
---------------	--

Returns

Minimum runtime in seconds,

max_elapsed()

```
double Bse::Test::Timer::max_elapsed ( ) const
```

Maximum time benchmarked for a *callee()* call.

min_elapsed()

```
double Bse::Test::Timer::min_elapsed ( ) const
```

Minimum time benchmarked for a *callee()* call.

n_reps()

```
int64 Bse::Test::Timer::n_reps ( ) const [inline]
```

Number of benchmark repetitions to execute.

test_elapsed()

```
double Bse::Test::Timer::test_elapsed ( ) const [inline]
```

Seconds spent in [benchmark\(\)](#)

The documentation for this class was generated from the following files:

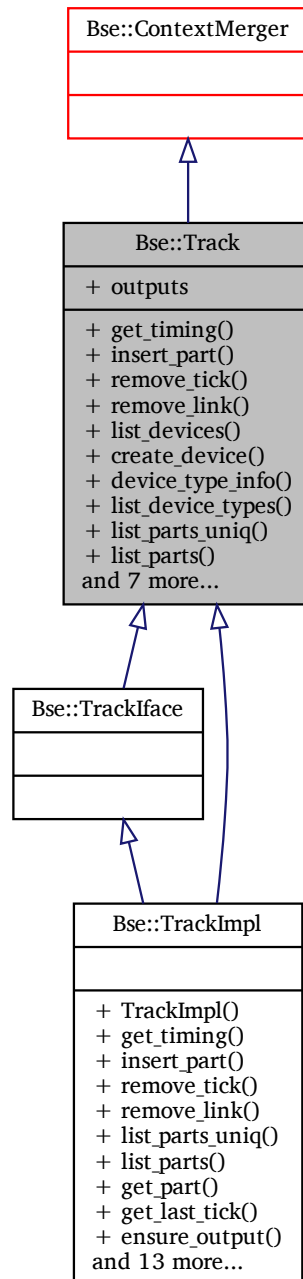
- bse/testing.hh
- bse/testing.cc

2.143 Bse::Track Interface Reference

Interface for sequencing information and links to [Part](#) objects.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Track:



Public Member Functions

- **SongTiming** `get_timing` ([int32](#) tick)
Retrieve song timing information at a specific tick.
- [int32](#) `insert_part` ([int32](#) tick, [Part](#) part)
Insert [Part](#) into [Track](#) at tick, returns the corresponding link id.
- `void remove_tick` ([int32](#) tick)
Remove [Part](#) at specified tick from a track.
- `void remove_link` ([int32](#) id)

- Remove a specific part link by ID from a track.*

 - `DeviceSeq list_devices ()`
List devices in order of processing.
 - `Device create_device (String device_id)`
Create a new device with device_type.
 - `DeviceInfo device_type_info (String device_id)`
Describe device_type.
 - `StringSeq list_device_types ()`
List known device types.
 - `PartSeq list_parts_uniq ()`
List all parts contained in a track.
 - `TrackPartSeq list_parts ()`
List parts scheduled in a track, sorted by tick.
 - `Part get_part (int32 tick)`
Get the part starting at a specific tick position.
 - `int32 get_last_tick ()`
Retrieve the last tick for this track.
 - `Error ensure_output ()`
Ensure the track has an output connection to a bus.
 - `Source get_output_source ()`
Get the output module of a track.

Public Attributes

- `ItemSeq outputs`
_(“Mixer busses used as output for this track.”)

Detailed Description

Interface for sequencing information and links to [Part](#) objects.

Member Function Documentation

create_device()

```
Device Bse::Track::create_device (
    String device_id )
```

Create a new device with device_type.

device_type_info()

```
DeviceInfo Bse::Track::device_type_info (
    String device_id )
```

Describe device_type.

ensure_output()

```
Error Bse::Track::ensure_output ( )
```

Ensure the track has an output connection to a bus.

get_last_tick()

`int32 Bse::Track::get_last_tick ()`
Retrieve the last tick for this track.

get_output_source()

`Source Bse::Track::get_output_source ()`
Get the output module of a track.
The output of this module is the merged result from all polyphonic voices and has all track specific alterations applied.

get_part()

`Part Bse::Track::get_part (`
 `int32 tick)`
Get the part starting at a specific tick position.

get_timing()

`SongTiming Bse::Track::get_timing (`
 `int32 tick)`
Retrieve song timing information at a specific tick.

insert_part()

`int32 Bse::Track::insert_part (`
 `int32 tick,`
 `Part part)`
Insert `Part` into `Track` at `tick`, returns the corresponding link id.

list_device_types()

`StringSeq Bse::Track::list_device_types ()`
List known device types.

list_devices()

`DeviceSeq Bse::Track::list_devices ()`
List devices in order of processing.

list_parts()

`TrackPartSeq Bse::Track::list_parts ()`
List parts scheduled in a track, sorted by tick.

list_parts_uniq()

`PartSeq Bse::Track::list_parts_uniq ()`
List all parts contained in a track.

remove_link()

```
void Bse::Track::remove_link (
    int32 id )
```

Remove a specific part link by ID from a track.

remove_tick()

```
void Bse::Track::remove_tick (
    int32 tick )
```

Remove [Part](#) at specified *tick* from a track.

Member Data Documentation

outputs

[ItemSeq](#) Bse::Track::outputs

_("Mixer busses used as output for this track.")

The documentation for this interface was generated from the following file:

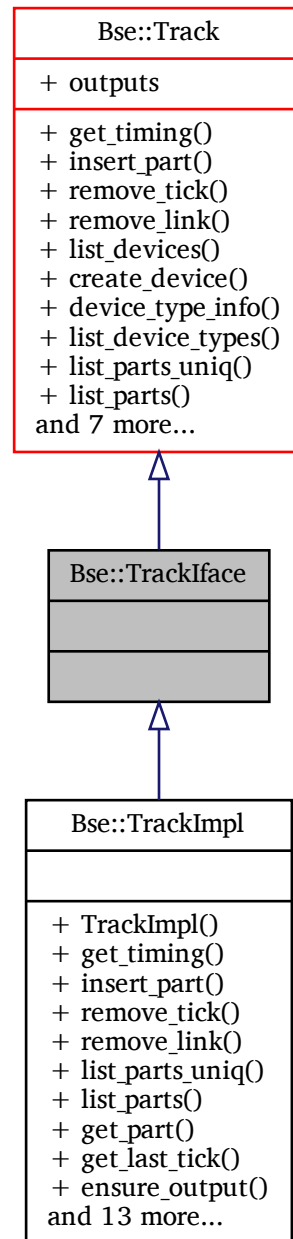
- [bse/bseapi.idl](#)

2.144 Bse::TrackIface Class Reference

IDL interface class for [Bse::Track](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::Trackiface:



Additional Inherited Members

Detailed Description

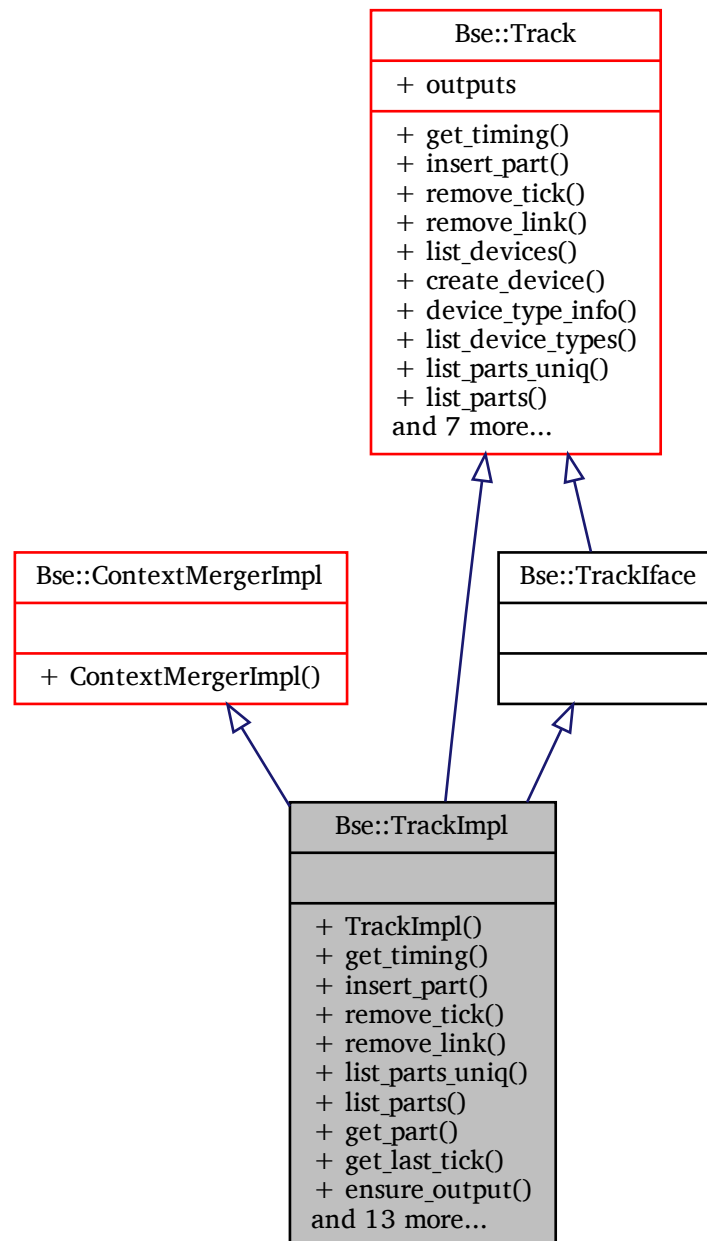
IDL interface class for [Bse::Track](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.145 Bse::TrackImpl Class Reference

Inheritance diagram for Bse::TrackImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- `bse/bsetrack.hh`
- `bse/bsetrack.cc`

2.146 Bse::TrackPart Struct Reference

Structure linking to a [Track](#) from within a [Part](#).

```
import "bseapi.idl";
```

Detailed Description

Structure linking to a [Track](#) from within a [Part](#).

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.147 Bse::TrackPartSeq Struct Reference

Sequence of [TrackPart](#) records.

```
import "bseapi.idl";
```

Detailed Description

Sequence of [TrackPart](#) records.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.148 Bse::TrackSeq Struct Reference

Sequence of [Track](#) objects.

```
import "bseapi.idl";
```

Detailed Description

Sequence of [Track](#) objects.

The documentation for this struct was generated from the following file:

- [bse/bseapi.idl](#)

2.149 Bse::ItemImpl::UndoDescriptor< Obj > Class Template Reference

[UndoDescriptor](#) - type safe object handle to persist undo/redo steps.

```
#include <bseitem.hh>
```

Detailed Description

```
template<class Obj>
```

```
class Bse::ItemImpl::UndoDescriptor< Obj >
```

[UndoDescriptor](#) - type safe object handle to persist undo/redo steps.

The documentation for this class was generated from the following file:

- [bse/bseitem.hh](#)

2.150 Bse::UserMessage Struct Reference

Structure for submission of user interface messages from BSE.

```
import "bseapi.idl";
```

Public Attributes

- [UserMessageType utype](#)
Severity classification for this message.
- [String title](#)
Usually GUI window title.
- [String text1](#)
Primary message to the user, should be limited to 80-100 chars.
- [String text2](#)
Explanatory (secondary) message no limitations recommended.
- [String text3](#)
Possibly (technical) details or machine error message.
- [String label](#)
Message class label, used to enable/disable this type of message.

Detailed Description

Structure for submission of user interface messages from BSE.

Member Data Documentation

label

`String Bse::UserMessage::label`

Message class label, used to enable/disable this type of message.

text1

`String Bse::UserMessage::text1`

Primary message to the user, should be limited to 80-100 chars.

text2

`String Bse::UserMessage::text2`

Explanatory (secondary) message no limitations recommended.

text3

`String Bse::UserMessage::text3`

Possibly (technical) details or machine error message.

title

`String Bse::UserMessage::title`

Usually GUI window title.

utype

`UserMessageType Bse::UserMessage::utype`

Severity classification for this message.

The documentation for this struct was generated from the following file:

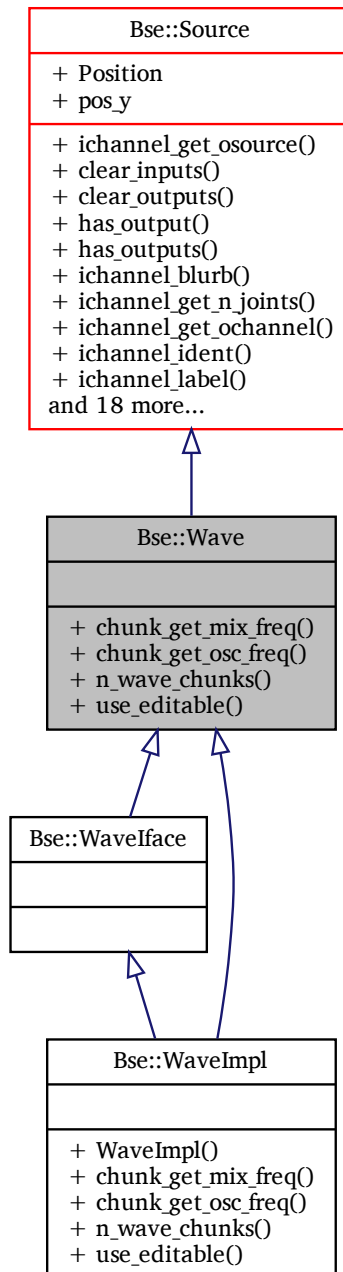
- [bse/bseapi.idl](#)

2.151 Bse::Wave Interface Reference

Interface for PCM wave samples.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::Wave:



Public Member Functions

- float64 `chunk_get_mix_freq` (int32 `chunk_index`)
Retrieve mixing frequency of a wave chunk.
- float64 `chunk_get_osc_freq` (int32 `chunk_index`)

Retrieve oscillating frequency of a wave chunk.

- [int32 n_wave_chunks \(\)](#)

Get the number of wave chunks of a wave.

- [EditableSample use_editable \(int32 chunk_index\)](#)

Retrieve an editable sample object for a wave chunk.

Detailed Description

Interface for PCM wave samples.

Member Function Documentation

chunk_get_mix_freq()

```
float64 Bse::Wave::chunk_get_mix_freq (
    int32 chunk_index )
```

Retrieve mixing frequency of a wave chunk.

chunk_get_osc_freq()

```
float64 Bse::Wave::chunk_get_osc_freq (
    int32 chunk_index )
```

Retrieve oscillating frequency of a wave chunk.

n_wave_chunks()

```
int32 Bse::Wave::n_wave_chunks ( )
```

Get the number of wave chunks of a wave.

use_editable()

```
EditableSample Bse::Wave::use_editable (
    int32 chunk_index )
```

Retrieve an editable sample object for a wave chunk.

The documentation for this interface was generated from the following file:

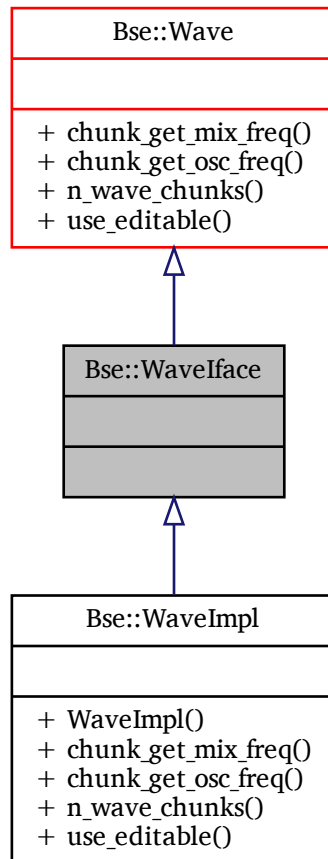
- [bse/bseapi.idl](#)

2.152 Bse::Waveface Class Reference

IDL interface class for [Bse::Wave](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::Waveface:



Additional Inherited Members

Detailed Description

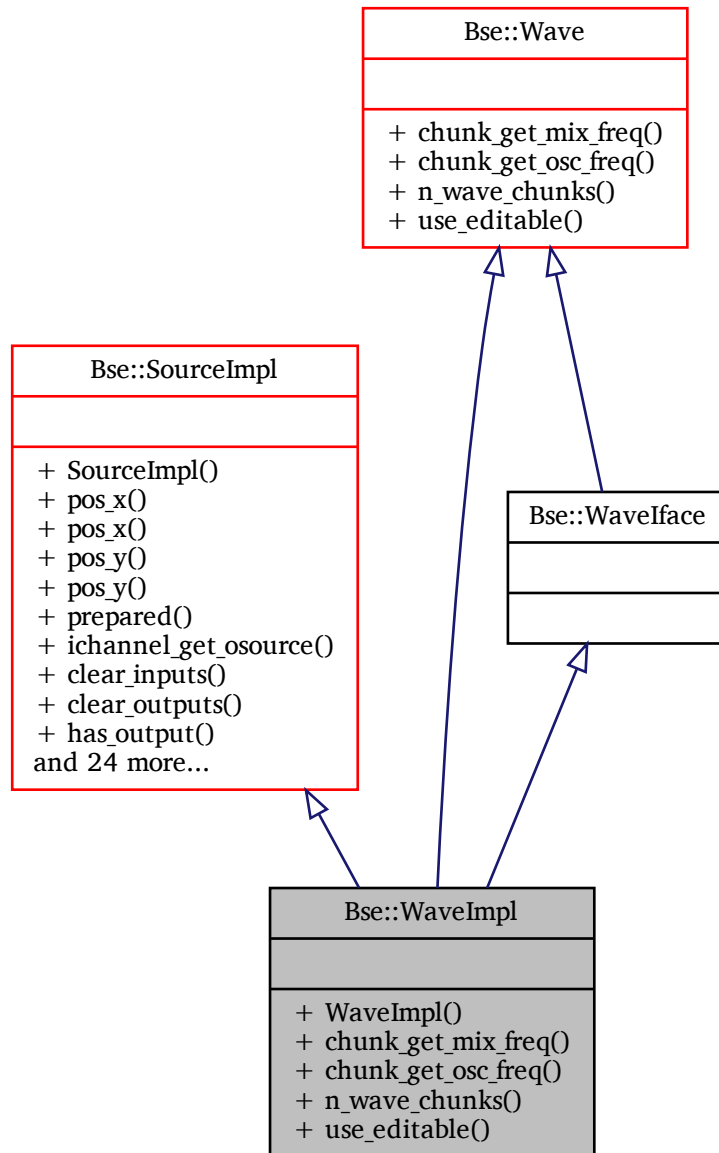
IDL interface class for [Bse::Wave](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.153 Bse::WaveImpl Class Reference

Inheritance diagram for Bse::WaveImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

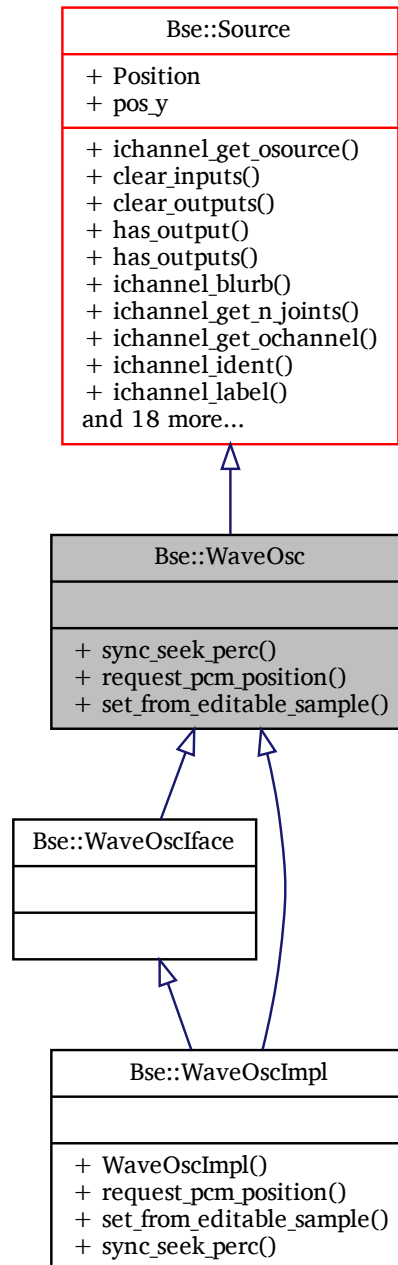
- bse/bsewave.hh
- bse/bsewave.cc

2.154 Bse::WaveOsc Interface Reference

Oscillator module for wave files.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::WaveOsc:



Public Member Functions

- `void sync_seek_perc` (float64 pos_perc, [WaveOscSeq](#) wosc_seq)

Seek a list of wave oscillators to a pcm position given in percentage. The oscillators will seek to the given position synchronously.

- void [request_pcm_position](#) ()
Request emission of the ::notify_pcm_position signal.
- void [set_from_editable_sample](#) ([EditableSample](#) esample)
Set wave to play from editable sample, bypassing undo and storage mechanisms.

Detailed Description

Oscillator module for wave files.

Member Function Documentation

[request_pcm_position\(\)](#)

```
void Bse::WaveOsc::request_pcm_position ( )
```

Request emission of the ::notify_pcm_position signal.

[set_from_editable_sample\(\)](#)

```
void Bse::WaveOsc::set_from_editable_sample (
    EditableSample esample )
```

Set wave to play from editable sample, bypassing undo and storage mechanisms.

[sync_seek_perc\(\)](#)

```
void Bse::WaveOsc::sync_seek_perc (
    float64 pos_perc,
    WaveOscSeq wosc_seq )
```

Seek a list of wave oscillators to a pcm position given in percentage. The oscillators will seek to the given position synchronously.

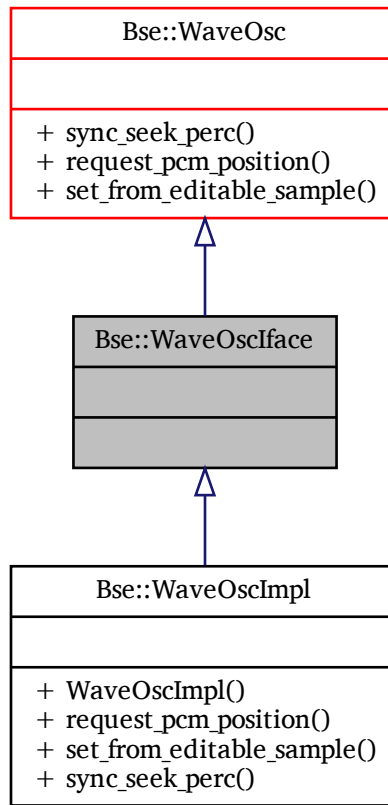
The documentation for this interface was generated from the following file:

- [bse/bseapi.idl](#)

2.155 Bse::WaveOscIface Class Reference

IDL interface class for [Bse::WaveOsc](#).
`#include <bseapi_interfaces.hh>`

Inheritance diagram for Bse::WaveOscIface:



Additional Inherited Members

Detailed Description

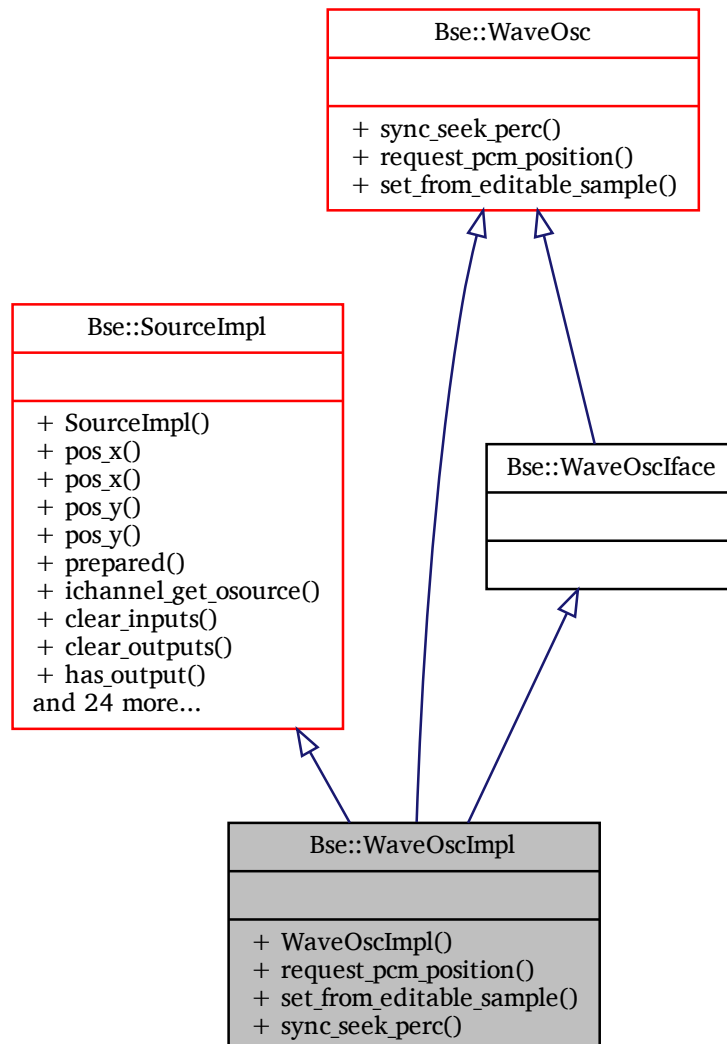
IDL interface class for [Bse::WaveOsc](#).

The documentation for this class was generated from the following file:

- `bse/bseapi_interfaces.hh`

2.156 Bse::WaveOscImpl Class Reference

Inheritance diagram for Bse::WaveOscImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- `bse/bsewaveosc.hh`
- `bse/bsewaveosc.cc`

2.157 Bse::WaveOscSeq Struct Reference

A list of part note events.

```
import "bseapi.idl";
```

Detailed Description

A list of part note events.

The documentation for this struct was generated from the following file:

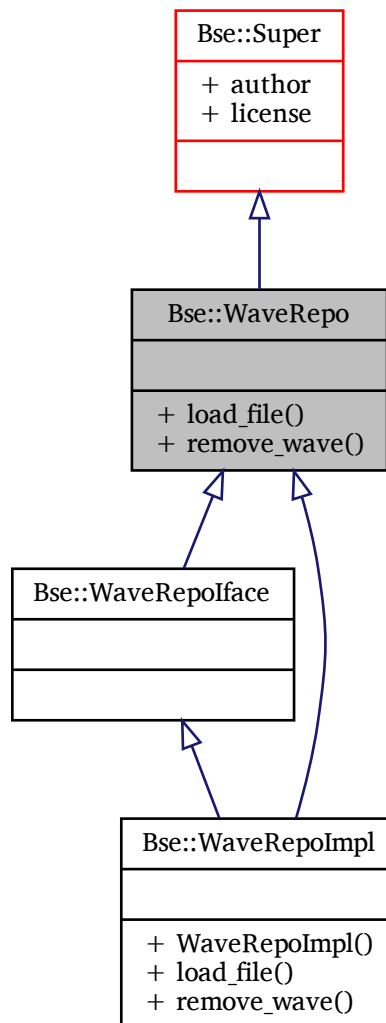
- [bse/bseapi.idl](#)

2.158 Bse::WaveRepo Interface Reference

Interface serving as container for [Wave](#) objects.

```
import "bseapi.idl";
```

Inheritance diagram for Bse::WaveRepo:



Public Member Functions

- Error `load_file` (`String` file_name)
Load wave from file.
- void `remove_wave` (`Wave` wave)
Remove a wave from repository.

Detailed Description

Interface serving as container for [Wave](#) objects.

Member Function Documentation

load_file()

```
Error Bse::WaveRepo::load_file (
    String file_name )
```

Load wave from file.

remove_wave()

```
void Bse::WaveRepo::remove_wave (
    Wave wave )
```

Remove a wave from repository.

The documentation for this interface was generated from the following file:

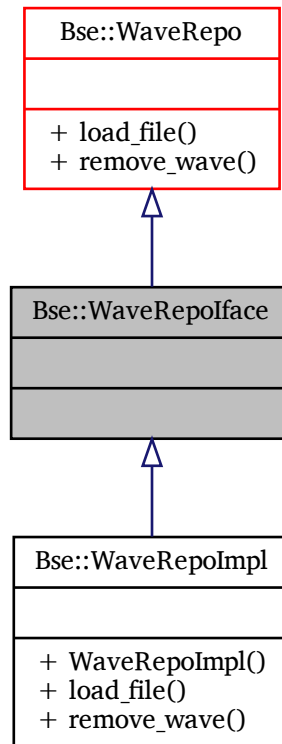
- [bse/bseapi.idl](#)

2.159 Bse::WaveRepoInterface Class Reference

IDL interface class for [Bse::WaveRepo](#).

```
#include <bseapi_interfaces.hh>
```

Inheritance diagram for Bse::WaveRepo:



Additional Inherited Members

Detailed Description

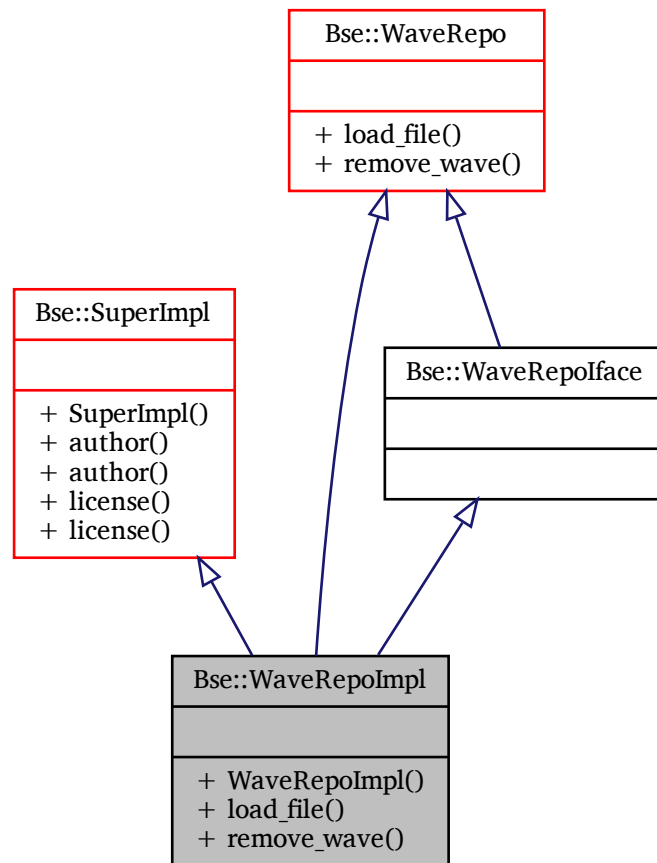
IDL interface class for [Bse::WaveRepo](#).

The documentation for this class was generated from the following file:

- bse/bseapi_interfaces.hh

2.160 Bse::WaveRepoImpl Class Reference

Inheritance diagram for Bse::WaveRepoImpl:



Additional Inherited Members

Detailed Description

The documentation for this class was generated from the following files:

- `bse/bsewaverepo.hh`
- `bse/bsewaverepo.cc`

Chapter 3

File Documentation

3.1 bse/bseapi.idl File Reference

Public BSE interface classes.

Classes

- struct [Bse::StringSeq](#)
Stringeq - a variable length list of test strings.
- struct [Bse::PixelSeq](#)
Representation of an image pixel sequence in ARGB format.
- struct [Bse::Icon](#)
Representation of an icon pixel image.
- struct [Bse::AuxData](#)
AuxData - record to describe entity attributes with "key = value" strings.
- struct [Bse::AuxDataSeq](#)
AuxDataSeq - a variable length list of *AuxData* records.
- struct [Bse::UserMessage](#)
Structure for submission of user interface messages from BSE.
- struct [Bse::SongTiming](#)
Song timing configuration.
- struct [Bse::NoteDescription](#)
A note description provides all needed details about a specific note. "
- interface [Bse::Object](#)
Base type for all new style C++ objects.
- interface [Bse::LegacyObject](#)
Base type for all legacy objects, derived from struct BseObject.
- struct [Bse::ItemSeq](#)
A list of Item or derived objects.
- struct [Bse::PropertyCandidates](#)
A list of items suitable to set as a specific property value.
- interface [Bse::Item](#)
Base interface type for objects that can be added to a container.
- struct [Bse::PartNote](#)
Part specific note event representation.
- struct [Bse::PartNoteSeq](#)
A list of part note events.
- struct [Bse::PartControl](#)
Part specific control event representation.
- struct [Bse::PartControlSeq](#)

- A list of part control events.*
- interface `Bse::Part`
 - Data interface for containment of piano notes and MIDI effects.*
- struct `Bse::PartSeq`
 - A list of `Part` or derived types.*
- struct `Bse::FloatSeq`
 - A list of floating point values.*
- struct `Bse::SharedMemory`
 - Descriptor for a shared memory region.*
- struct `Bse::ProbeFeatures`
 - Bits representing a selection of probe sample data features.*
- interface `Bse::SignalMonitor`
 - Interface for monitoring output signals.*
- interface `Bse::Source`
 - Base interface type for synthesis modules with input or output streams.*
- interface `Bse::Container`
 - Base interface type for containers of `Item` derived types.*
- interface `Bse::ContextMerger`
 - Source module for merging multiple synthesis contexts, used to implement polyphony.*
- interface `Bse::Super`
 - Base interface type for `Item` managers.*
- struct `Bse::SuperSeq`
 - A list of `Super` type objects.*
- interface `Bse::SNet`
 - Base interface type for all kinds of synthesis networks.*
- interface `Bse::CSynth`
 - Customizable synthesis (filter) network container.*
- interface `Bse::SubSynth`
 - Synthesizer module for embedding (rerouting input and output) of another synthesizer network (`SNet`).*
- struct `Bse::ModuleTypeInfo`
 - Info for module types.*
- class `Bse::Module`
 - Interface for the encapsulation of audio processors.*
- struct `Bse::DeviceTypeInfo`
 - Info for device types.*
- interface `Bse::Device`
 - Interface for the encapsulation of audio processors.*
- interface `Bse::Track`
 - Interface for sequencing information and links to `Part` objects.*
- struct `Bse::TrackSeq`
 - Sequence of `Track` objects.*
- struct `Bse::PartLink`
 - Record representing the use of a `Part` within a `Track` at a specific position.*
- struct `Bse::PartLinkSeq`
 - Sequence of `PartLink` records.*
- struct `Bse::TrackPart`
 - Structure linking to a `Track` from within a `Part`.*
- struct `Bse::TrackPartSeq`
 - Sequence of `TrackPart` records.*
- interface `Bse::Bus`
 - Interface for effect stacks and per-track audio signal routing to the master output.*
- interface `Bse::Song`

- Interface for **Track** and **Part** objects, as well as meta data for sequencing.*

 - struct **Bse::SampleFileInfo**
Structure containing meta data for multi wave samples.
 - interface **Bse::EditableSample**
Interface for editable PCM wave samples.
 - interface **Bse::Wave**
Interface for PCM wave samples.
 - interface **Bse::WaveRepo**
*Interface serving as container for **Wave** objects.*
 - struct **Bse::WaveOscSeq**
A list of part note events.
 - interface **Bse::WaveOsc**
Oscillator module for wave files.
 - interface **Bse::SoundFont**
Interface for sound fonts.
 - interface **Bse::MidiNotifier**
Interface for MIDI event notification.
 - interface **Bse::MidiSynth**
Interface for MIDI synthesis networks.
 - interface **Bse::Project**
Projects support loading, saving, playback and act as containers for all other sound objects.
 - interface **Bse::PcmWriter**
Interface for writing PCM wave data.
 - struct **Bse::Category**
Categories describe useful type entities.
 - struct **Bse::CategorySeq**
*Sequence of **Category** records.*
 - struct **Bse::DriverEntry**
Driver information for PCM and MIDI handling.
 - struct **Bse::DriverEntrySeq**
***DriverEntry** sequence.*
 - struct **Bse::ShmFragment**
Fragment description for interesting bits of shared memory.
 - struct **Bse::ShmFragmentSeq**
Collection of shared memory fragments.
 - interface **Bse::Server**
*Main **Bse** remote origin object.*

Modules

- **Bse**
*The **Bse** namespace contains all functions of the synthesis engine.*

Enumerations

- enum **Bse::MonitorField** {
Bse::F64_GENERATION, **Bse::F32_MIN**, **Bse::F32_MAX**, **Bse::F32_DB_SPL**,
Bse::F32_DB_TIP, **Bse::END_BYTE** }
- Offsets for signal monitoring fields in bytes, field type and size is used as prefix.*
- enum **Bse::UserMessageType** { **Bse::ERROR**, **Bse::WARNING**, **Bse::INFO**, **Bse::DEBUG** }
- enum **Bse::SongTelemetry** { **Bse::I32_TICK_POINTER**, **Bse::BYTECOUNT** }
- Offsets for signal monitoring fields in bytes, field type and size is used as prefix.*
- enum **Bse::ProjectState** { **Bse::INACTIVE**, **Bse::ACTIVE**, **Bse::PLAYING** }
- Enumeration describing the current activation and playback state of a project.*

Variables

- Const `Bse::KAMMER_NOTE`
Value represents unparsable/unknown notes.
- Const `Bse::KAMMER_FREQ`
Kammer note, representing the kammer frequency's MIDI note value for A' or A4.
- Const `Bse::KAMMER_OCTAVE`
Pitch Standard, see also: [https://en.wikipedia.org/wiki/A440_\(pitch_standard\)](https://en.wikipedia.org/wiki/A440_(pitch_standard))
- Const `Bse::MIN_OCTAVE`
Octave number for MIDI A'.
- Const `Bse::MAX_OCTAVE`
Octave of MIN_NOTE.
- Const `Bse::MIN_FINE_TUNE`
Octave of MAX_NOTE.

Detailed Description

Public BSE interface classes.

Bibliography

- [Bellare und Yee 2001] BELLARE, Mihir ; YEE, Bennet: *Forward-Security in Private-Key Cryptography*. Cryptology ePrint Archive, Report 2001/035. 2001. – URL <https://eprint.iacr.org/2001/035> 110
- [Division 2014] DIVISION, NIST Computer S.: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions / National Institute of Standards and Technology, U.S. Department of Commerce. URL http://csrc.nist.gov/publications/drafts/fips-202/fips_202_draft.pdf, May 2014 (202). – FIPS Publication 178, 179, 180, 181, 182, 183
- [Peeters und Assche 2011] PEETERS, G. Bertoni, J. Daemen, M. ; ASSCHE, G. V.: *The Keccak reference*. Round 3 submission to NIST SHA-3. 2011. – URL <http://keccak.noekeon.org/Keccak-reference-3.0.pdf> 49, 104, 105, 109
- [Swanson 2013] SWANSON, K. Mowery, M. Wei, D. Kohlbrenner, H. Shacham, S.: Welcome to the Entropics: Boot-Time Entropy in Embedded Devices. In: *2013 IEEE Symposium on Security and Privacy* (2013), June, S. 589–603. – URL <http://cseweb.ucsd.edu/~hovav/dist/earlyentropy.pdf> 25

Index

- - Bse, 23
- ~KeccakRng
 - Bse::KeccakRng, 110
- abspath
 - Bse::Path, 50
- ac_stamp
 - Bse::TaskStatus, 224
- activate
 - Bse::Project, 149
- add
 - Bse::TaskRegistry, 222
- aligned_alloc
 - Bse, 23
- aligned_free
 - Bse, 23
- allocate_aligned_block
 - Bse, 23
- application_name
 - Bse, 23
- application_name_init
 - Bse, 24
- apply_idl_property
 - Bse::ItemImpl, 101
- as_hex
 - Bse::Xms::SerializationField, 162
- as_node
 - Bse::Xms::SerializationField, 162
- attribute
 - Bse::Xms::SerializationField, 162
- attributes
 - Bse::AuxData, 60
- auto_deactivate
 - Bse::Project, 149
- auto_seed
 - Bse::KeccakRng, 110
 - Bse::Pcg32Rng, 142
- basename
 - Bse::Path, 50
- beastbse_cachedir_cleanup
 - Bse, 24
- beastbse_cachedir_create
 - Bse, 24
- beastbse_cachedir_current
 - Bse, 24
- benchmark
 - Bse::Test::Timer, 226
- binary_lookup
 - Bse, 24
- binary_lookup_insertion_pos
 - Bse, 24
- binary_lookup_sibling
 - Bse, 24
- bit_capacity
 - Bse::KeccakRng, 110
- Blob
 - Bse::Blob, 61
- breakpoint
 - Bse, 25
- broadcast_shm_fragments
 - Bse::Server, 170
- Bse, 6
 - _, 23
 - aligned_alloc, 23
 - aligned_free, 23
 - allocate_aligned_block, 23
 - application_name, 23
 - application_name_init, 24
 - beastbse_cachedir_cleanup, 24
 - beastbse_cachedir_create, 24
 - beastbse_cachedir_current, 24
 - binary_lookup, 24
 - binary_lookup_insertion_pos, 24
 - binary_lookup_sibling, 24
 - breakpoint, 25
 - cached_hash_secret, 47
 - collect_runtime_entropy, 25
 - collect_system_entropy, 25
 - constexpr_equals, 25
 - copy_reordered, 25
 - cpu_arch, 26
 - cpu_info, 26
 - create_memory_area, 26
 - cstrings_to_vector, 26
 - current_locale_strtold, 26
 - debug, 26
 - debug_enabled, 26
 - debug_key_enabled, 27
 - debug_key_value, 27
 - debug_message, 27
 - delete_inplace, 27
 - diag_abort_hook, 27
 - exec_handler_clear, 27
 - exec_now, 27, 28
 - exec_timeout, 28
 - executable_name, 28
 - executable_path, 28
 - fatal_error, 28
 - feature_check, 28
 - feature_toggle_bool, 29
 - feature_toggle_find, 29
 - find_memory_area, 29
 - fmsb, 29
 - icon_from_pixstream, 29
 - icon_sanitize, 29
 - info, 29
 - init_async, 30
 - init_glue_context, 30
 - init_needed, 30
 - init_server_instance, 30
 - int16, 20
 - int32, 20
 - int64, 20
 - int8, 20
 - KAMMER_FREQ, 47
 - KAMMER_NOTE, 47
 - KAMMER_OCTAVE, 47
 - MAX_OCTAVE, 47
 - MIN_FINE_TUNE, 47
 - MIN_OCTAVE, 47
 - memset4, 30
 - ModuleFlag, 21
 - MonitorField, 21
 - monotonic_counter, 30
 - new_inplace, 30
 - posix_locale_strtold, 31
 - print_backtrace, 31
 - printerr, 31
 - printout, 31
 - program_alias, 31
 - program_alias_init, 31
 - program_cwd, 31
 - ProjectState, 22
 - random_float, 32
 - random_frange, 32
 - random_int64, 32
 - random_irange, 32
 - random_nonce, 32
 - random_secret, 32
 - release_aligned_block, 32
 - runpath, 32
 - sha3_224_hash, 33
 - sha3_256_hash, 33
 - sha3_384_hash, 33
 - sha3_512_hash, 33
 - shake128_hash, 33
 - shake256_hash, 33
 - shared_ptr_cast, 34
 - SongTelemetry, 22
 - String, 20
 - string_canonify, 34
 - string_capitalize, 35
 - string_casecmp, 35
 - string_casefold, 35
 - string_cmp, 35
 - string_cmp_uuid, 35
 - string_endswith, 35
 - string_format, 35
 - string_from_bool, 35
 - string_from_cquote, 36
 - string_from_double, 36
 - string_from_double_vector, 36
 - string_from_errno, 36
 - string_from_float, 36
 - string_from_int, 36
 - string_from_pretty_function_name, 36
 - string_from_type, 36
 - string_from_uint, 37
 - string_from_unicode, 37
 - string_has_int, 37
 - string_hexdump, 37
 - string_is_canonified, 37
 - string_is_uuid, 37
 - string_join, 37
 - string_locale_format, 38
 - string_locale_vprintf, 38
 - string_lstrip, 38
 - string_match_identifier, 38
 - string_match_identifier_tail, 38
 - string_multiply, 38
 - string_normalize_nfc, 38
 - string_normalize_nfd, 39
 - string_normalize_nfkc, 39
 - string_normalize_nfkd, 39
 - string_option_check, 39
 - string_option_get, 39

- string_options_split, 39
- string_replace, 39
- string_rstrip, 40
- string_set_A2Z, 40
- string_set_a2z, 40
- string_set_ascii_alnum, 40
- string_split, 40
- string_split_any, 40
- string_startswith, 40
- string_strip, 40
- string_substitute_char, 40
- string_to_bool, 41
- string_to_cescape, 41
- string_to_cquote, 41
- string_to_double, 41
- string_to_double_vector, 41
- string_to_int, 41
- string_to_type, 42
- string_to_uint, 42
- string_tolower, 42
- string_totitle, 42
- string_toupper, 42
- string_vector_erase_empty, 42
- string_vector_find, 42
- string_vector_find_value, 43
- string_vector_lstrip, 43
- string_vector_rstrip, 43
- string_vector_strip, 43
- string_vprintf, 43
- StringVector, 20
- text_convert, 43
- timestamp_benchmark, 44
- timestamp_format, 44
- timestamp_realtime, 44
- timestamp_resolution, 44
- timestamp_startup, 44
- uint, 21
- uint16, 21
- uint32, 21
- uint64, 21
- uint8, 21
- unichar, 21
- unicode_is_assigned, 44
- unicode_is_character, 44
- unicode_is_control_code, 44
- unicode_is_noncharacter, 44
- unicode_is_private, 45
- unicode_is_valid, 45
- url_show, 45
- UserMessageType, 22
- utf8_to_unicode, 45
- utf8len, 45
- vector_erase_element, 46
- vector_erase_iface, 46
- version, 46
- warning, 46
- zintern_decompress, 46
- zintern_free, 47
- bse/bseapi.idl, 247
- Bse::AlignedArray< T, ALIGNMENT >, 58
- Bse::AlignedPOD< SIZE >, 58
- Bse::AnsiColors, 47
 - color, 48
 - color_code, 48
 - colorize_tty, 48
 - Colors, 48
 - configure, 49
- Bse::AsyncBlockingQueue< Value >, 58
- Bse::AutoSeeder, 59
 - generate, 59
 - operator(), 59
 - random, 59
- Bse::AuxData, 60
 - attributes, 60
 - entity, 60
- Bse::AuxDataSeq, 60
- Bse::Blob, 60
 - Blob, 61
 - bytes, 61
 - data, 61
 - from_file, 62
 - from_url, 62
 - name, 62
 - operator bool, 62
 - size, 62
 - string, 62
- Bse::Bus, 62
 - connect_bus, 64
 - connect_track, 64
 - disconnect_bus, 64
 - disconnect_track, 64
 - ensure_output, 64
- Bse::BusIface, 64
- Bse::BusImpl, 66
- Bse::CSynth, 74
- Bse::CSynthIface, 75
- Bse::CSynthImpl, 76
- Bse::Category, 66
- Bse::CategorySeq, 67
- Bse::Container, 67
 - get_item, 68
 - list_children, 68
 - lookup_item, 68
- Bse::ContainerIface, 68
- Bse::ContainerImpl, 70
- Bse::ContextMerger, 70
- Bse::ContextMergerIface, 71
- Bse::ContextMergerImpl, 73
- Bse::DataKey
 - destroy, 77
 - fallback, 77
- Bse::DataKey< Type >, 77
- Bse::DataList, 77
- Bse::DataListContainer, 78
 - delete_data, 79
 - get_data, 79
 - set_data, 79
 - swap_data, 79
- Bse::Device, 80
 - create_module, 81
 - device_type_info, 81
 - get_device_type, 81
 - list_module_types, 81
 - list_modules, 81
 - module_type_info, 81
- Bse::DeviceCrawlerIface, 82
- Bse::DeviceCrawlerImpl, 83
- Bse::DeviceIface, 83
 - Bse::DeviceImpl, 85
 - Bse::DeviceInfo, 86
 - Bse::DriverEntry, 86
 - Bse::DriverEntrySeq, 86
 - Bse::EditableSample, 86
 - close, 88
 - collect_stats, 88
 - get_length, 88
 - get_n_channels, 88
 - get_osc_freq, 88
 - open, 89
 - Bse::EditableSampleIface, 89
 - Bse::EditableSampleImpl, 90
 - Bse::Flac1Handle, 91
 - create, 91
 - put_wstore, 91
 - read_data, 91
 - Bse::FloatSeq, 92
 - Bse::FriendAllocator
 - construct, 93
 - destroy, 93
 - make_shared, 93
 - Bse::FriendAllocator< T >, 92
 - Bse::Icon, 93
 - Bse::Item, 94
 - check_is_a, 95
 - common_ancestor, 95
 - editable_property, 95
 - get_name, 96
 - get_name_or_type, 96
 - get_parent, 96
 - get_project, 96
 - get_property_candidates, 96
 - get_seqid, 96
 - get_type, 96
 - get_type_authors, 96
 - get_type_blurb, 96
 - get_type_license, 97
 - get_type_name, 97
 - get_undef_path, 97
 - group_undo, 97
 - internal, 97
 - set_name, 97
 - ungroup_undo, 97
 - unuse, 97
 - use, 97
 - Bse::ItemIface, 98
 - Bse::ItemImpl, 100
 - apply_idl_property, 101
 - push_property_undo, 101
 - push_undo, 101
 - push_undo_to_redo, 102
 - undo_descriptor, 102
 - undo_resolve, 102
 - Bse::ItemImpl::UndoDescriptor< Obj >, 233
 - Bse::ItemSeq, 102
 - Bse::KeccakCryptoRng, 102
 - KeccakCryptoRng, 103
 - Bse::KeccakFastRng, 105
 - KeccakFastRng, 106
 - Bse::KeccakGoodRng, 106
 - KeccakGoodRng, 107
 - Bse::KeccakRng, 108
 - ~KeccakRng, 110

- auto_seed, 110
- bit_capacity, 110
- discard, 110
- forget, 110
- generate, 110
- KeccakRng, 110
- max, 110
- min, 110
- n_nums, 111
- operator!, 111
- operator<<, 112
- operator>>, 112
- operator(), 111
- operator = =, 112
- random, 111
- result_type, 109
- seed, 111
- xor_seed, 111
- Bse::LegacyObject, 112
 - debug_name, 114
 - find_typedata, 114
 - proxy_id, 114
 - unique_id, 114
- Bse::LegacyObjectIface, 114
- Bse::LegacyObjectImpl, 116
- Bse::Lib, 49
- Bse::Lib::KeccakF1600, 104
 - byte, 104
 - KeccakF1600, 104
 - permute, 104
 - reset, 105
- Bse::Lib::ScopedLocale, 159
- Bse::Lib::ScopedPosixLocale, 159
 - posix_locale, 160
- Bse::Lib::StringFormatter, 213
 - format, 213
- Bse::MidiNotifier, 116
- Bse::MidiNotifierIface, 118
- Bse::MidiNotifierImpl, 119
- Bse::MidiSynth, 119
- Bse::MidiSynthIface, 121
- Bse::MidiSynthImpl, 122
- Bse::Module, 122
 - get_module_type, 124
 - module_type_info, 124
- Bse::ModuleIface, 124
- Bse::ModuleImpl, 126
- Bse::ModuleTypeInfo, 127
- Bse::NoteDescription, 127
- Bse::Object, 127
 - find_prop, 128
 - get_prop, 128
 - list_props, 128
 - notify, 128
 - set_prop, 128
- Bse::ObjectIface, 129
- Bse::ObjectImpl, 130
- Bse::Part, 130
 - change_control, 133
 - change_note, 133
 - check_overlap, 133
 - delete_event, 133
 - deselect_controls, 133
 - deselect_event, 133
 - deselect_notes, 133
 - get_channel_controls, 134
 - get_controls, 134
 - get_last_tick, 134
 - get_max_note, 134
 - get_min_note, 134
 - get_notes, 134
 - get_timing, 134
 - insert_control, 134
 - insert_note, 135
 - insert_note_auto, 135
 - is_event_selected, 135
 - list_controls, 135
 - list_links, 135
 - list_notes_crossing, 135
 - list_notes_within, 135
 - list_selected_controls, 136
 - list_selected_notes, 136
 - queue_controls, 136
 - queue_notes, 136
 - select_controls, 136
 - select_controls_exclusive, 136
 - select_event, 136
 - select_notes, 137
 - select_notes_exclusive, 137
- Bse::PartControl, 137
- Bse::PartControlSeq, 137
- Bse::PartIface, 138
- Bse::PartImpl, 139
- Bse::PartLink, 140
- Bse::PartLinkSeq, 140
- Bse::PartNote, 140
- Bse::PartNoteSeq, 140
- Bse::PartSeq, 140
- Bse::Path, 49
 - abspath, 50
 - basename, 50
 - cache_home, 51
 - check, 51
 - config_dirs, 51
 - config_home, 51
 - config_names, 51, 52
 - cwd, 52
 - data_dirs, 52
 - data_home, 52
 - dirname, 52
 - equals, 52
 - expand_tilde, 52
 - isabs, 52
 - isdirname, 53
 - mkdirs, 53
 - realpath, 53
 - runtime_dir, 53
 - searchpath_contains, 53
 - searchpath_find, 53
 - searchpath_list, 53
 - searchpath_multiply, 53
 - user_home, 54
- Bse::Pcg32Rng, 141
 - auto_seed, 142
 - Pcg32Rng, 141, 142
 - random, 142
 - seed, 142
- Bse::PcmWriter, 142
- Bse::PcmWriterIface, 144
- Bse::PcmWriterImpl, 145
- Bse::PixelSeq, 145
- Bse::ProbeFeatures, 146
- Bse::Procedure, 54
- Bse::Project, 146
 - activate, 149
 - auto_deactivate, 149
 - can_play, 149
 - change_name, 149
 - clean_dirty, 149
 - clear_undo, 149
 - create_csynth, 149
 - create_midi_synth, 150
 - create_song, 150
 - deactivate, 150
 - get_midi_notifier, 150
 - get_sound_font_repo, 150
 - get_state, 150
 - get_supers, 150
 - get_wave_repo, 150
 - import_midi_file, 150
 - inject_midi_control, 151
 - is_active, 151
 - is_dirty, 151
 - is_playing, 151
 - play, 151
 - redo, 151
 - redo_depth, 151
 - remove_snet, 151
 - restore_from_file, 151
 - start_playback, 152
 - State, 152
 - stop, 152
 - stop_playback, 152
 - store, 152
 - store_bse, 152
 - undo, 152
 - undo_depth, 152
- Bse::ProjectIface, 153
- Bse::ProjectImpl, 155
- Bse::PropertyCandidates, 156
- Bse::Re, 54
- Bse::Resampler2, 156
 - delay, 157
 - find_precision_for_bits, 157
 - order, 157
 - precision_name, 157
 - process_block, 158
 - Resampler2, 157
 - reset, 158
 - sse_available, 158
 - sse_enabled, 158
 - test_filter_impl, 158
- Bse::SHA3_224, 178
 - digest, 179
 - reset, 179
 - SHA3_224, 178
 - update, 179
- Bse::SHA3_256, 179
 - digest, 180
 - reset, 180
 - SHA3_256, 179
 - update, 180
- Bse::SHA3_384, 180
 - digest, 181
 - reset, 181

- SHA3_384, 180
- update, 181
- Bse::SHA3_512, 181
- digest, 182
- reset, 182
- SHA3_512, 181
- update, 182
- Bse::SHAKE128, 182
- reset, 183
- SHAKE128, 182
- squeeze_digest, 183
- update, 183
- Bse::SHAKE256, 183
- reset, 184
- SHAKE256, 183
- squeeze_digest, 184
- update, 184
- Bse::SNet, 189
- can_create_source, 190
- create_source, 191
- remove_source, 191
- supports_user_synths, 191
- Bse::SNetIface, 191
- Bse::SNetImpl, 193
- Bse::SampleFileInfo, 158
- Bse::Sequencer, 160
- Bse::Server, 167
- broadcast_shm_fragments, 170
- can_load, 170
- category_match, 171
- category_match_typed, 171
- create_project, 171
- destroy_project, 171
- engine_active, 171
- find_module_type, 171
- from_proxy, 171
- get_config, 171
- get_config_defaults, 171
- get_custom_effect_dir, 172
- get_custom_instrument_dir, 172
- get_demo_path, 172
- get_effect_path, 172
- get_instrument_path, 172
- get_ladspa_path, 172
- get_mp3_version, 172
- get_plugin_path, 172
- get_sample_path, 172
- get_shared_memory, 172
- get_version, 173
- get_vorbis_version, 173
- last_project, 173
- list_midi_drivers, 173
- list_module_types, 173
- list_pcm_drivers, 173
- load_assets, 173
- locked_config, 173
- module_type_icon, 173
- note_construct, 173
- note_describe, 174
- note_from_freq, 174
- note_from_string, 174
- note_to_freq, 174
- purge_stale_cachedirs, 174
- register_core_plugins, 174
- register_ladspa_plugins, 174
- sample_file_info, 174
- send_user_message, 175
- set_config, 175
- start_recording, 175
- test_counter_inc_fetch, 175
- tick_stamp_from_systime, 175
- Bse::ServerIface, 175
- Bse::ServerImpl, 177
- register_source_module, 178
- Bse::SharedMemory, 184
- shm_creator, 184
- shm_length, 185
- shm_start, 185
- Bse::ShmFragment, 185
- Bse::ShmFragmentSeq, 185
- Bse::SignalMonitor, 185
- get_frame_duration, 187
- get_mix_freq, 187
- get_ochannel, 187
- get_osource, 187
- get_probe_features, 187
- get_shm_offset, 187
- set_probe_features, 187
- Bse::SignalMonitorIface, 188
- Bse::SignalMonitorImpl, 189
- Bse::Song, 194
- create_bus, 195
- create_part, 195
- create_track, 195
- ensure_master_bus, 195
- ensure_track_links, 196
- find_any_track_for_part, 196
- find_track_for_part, 196
- get_master_bus, 196
- get_shm_offset, 196
- get_timing, 196
- list_tracks, 196
- remove_bus, 196
- remove_part, 196
- remove_track, 197
- synthesize_note, 197
- Bse::SongIface, 197
- Bse::SongImpl, 199
- Bse::SongTiming, 200
- Bse::SoundFont, 200
- Bse::SoundFontIface, 201
- Bse::SoundFontImpl, 202
- Bse::SoundFontRepoIface, 202
- Bse::SoundFontRepoImpl, 204
- Bse::Source, 204
- clear_inputs, 206
- clear_outputs, 206
- create_signal_monitor, 207
- get_automation_channel, 207
- get_automation_control, 207
- get_mix_freq, 207
- has_output, 207
- has_outputs, 207
- ichannel_blurb, 207
- ichannel_get_n_joints, 207
- ichannel_get_ochannel, 207
- ichannel_get_osource, 208
- ichannel_ident, 208
- ichannel_label, 208
- is_joint_ichannel, 208
- is_joint_ichannel_by_id, 208
- is_prepared, 208
- n_ichannels, 208
- n_ochannels, 208
- ochannel_blurb, 208
- ochannel_ident, 209
- ochannel_label, 209
- set_automation, 209
- set_input, 209
- set_input_by_id, 209
- set_pos, 209
- unset_input, 210
- unset_input_by_id, 210
- Bse::SourceIface, 210
- Bse::SourceImpl, 212
- Bse::Spinlock, 212
- Bse::StringSeq, 215
- Bse::Strings, 214
- Bse::SubSynth, 215
- Bse::SubSynthIface, 217
- Bse::SubSynthImpl, 218
- Bse::Super, 219
- Bse::SuperIface, 219
- Bse::SuperImpl, 221
- Bse::SuperSeq, 221
- Bse::TaskRegistry, 222
- add, 222
- list, 222
- remove, 222
- update, 222
- Bse::TaskStatus, 223
- ac_stamp, 224
- cstime, 224
- cutime, 224
- name, 224
- priority, 224
- process_id, 224
- processor, 224
- state, 224
- stime, 225
- string, 224
- task_id, 225
- TaskStatus, 223
- update, 224
- utime, 225
- Bse::Test, 54
- init, 55
- random_float, 55
- random_frange, 55
- random_int64, 55
- random_irange, 55
- run, 55, 56
- slow, 56
- stringify_arg, 56
- verbose, 56
- Bse::Test::Timer, 225
- benchmark, 226
- max_elapsed, 226
- min_elapsed, 226
- n_reps, 226
- test_elapsed, 226
- Timer, 225
- Bse::Track, 226
- create_device, 228
- device_type_info, 228

- ensure_output, 228
- get_last_tick, 228
- get_output_source, 229
- get_part, 229
- get_timing, 229
- insert_part, 229
- list_device_types, 229
- list_devices, 229
- list_parts, 229
- list_parts_uniq, 229
- outputs, 230
- remove_link, 229
- remove_tick, 230
- Bse::TrackIface, 230
- Bse::TrackImpl, 232
- Bse::TrackPart, 233
- Bse::TrackPartSeq, 233
- Bse::TrackSeq, 233
- Bse::UserMessage, 233
 - label, 234
 - text1, 234
 - text2, 234
 - text3, 234
 - title, 234
 - utype, 234
- Bse::Wave, 235
 - chunk_get_mix_freq, 236
 - chunk_get_osc_freq, 236
 - n_wave_chunks, 236
 - use_editable, 236
- Bse::WaveIface, 236
- Bse::WaveImpl, 238
- Bse::WaveOsc, 239
 - request_pcm_position, 240
 - set_from_editable_sample, 240
 - sync_seek_perc, 240
- Bse::WaveOscIface, 240
- Bse::WaveOscImpl, 242
- Bse::WaveOscSeq, 242
- Bse::WaveRepo, 243
 - load_file, 244
 - remove_wave, 244
- Bse::WaveRepoIface, 244
- Bse::WaveRepoImpl, 246
- Bse::Xms, 56
- Bse::Xms::DataConverter< T, typename >, 76
- Bse::Xms::Reflink, 156
- Bse::Xms::SerializableInterface, 161
- Bse::Xms::SerializationField, 161
 - as_hex, 162
 - as_node, 162
 - attribute, 162
 - hex, 162
 - node, 162
 - operator &, 162
 - serialization_node, 162
- Bse::Xms::SerializationNode, 163
 - children, 164
 - create_child, 164
 - first_child, 164
 - get, 164
 - has, 165
 - in_load, 165
 - in_save, 165
 - load, 165
 - loading, 165
 - name, 165
 - null_id, 166
 - operator bool, 165
 - operator[], 165
 - parse_xml, 165
 - reflink, 166
 - save, 166
 - save_under, 166
 - SerializationNode, 164
 - with_default, 166
 - write_xml, 166
- Bse::Xms::SerializationNode::←
 - QueuedArgs, 156
- byte
 - Bse::Lib::KeccakF1600, 104
- bytes
 - Bse::Blob, 61
- cache_home
 - Bse::Path, 51
- cached_hash_secret
 - Bse, 47
- can_create_source
 - Bse::SNet, 190
- can_load
 - Bse::Server, 170
- can_play
 - Bse::Project, 149
- category_match
 - Bse::Server, 171
- category_match_typed
 - Bse::Server, 171
- change_control
 - Bse::Part, 133
- change_name
 - Bse::Project, 149
- change_note
 - Bse::Part, 133
- check
 - Bse::Path, 51
- check_is_a
 - Bse::Item, 95
- check_overlap
 - Bse::Part, 133
- children
 - Bse::Xms::SerializationNode, 164
- chunk_get_mix_freq
 - Bse::Wave, 236
- chunk_get_osc_freq
 - Bse::Wave, 236
- clean_dirty
 - Bse::Project, 149
- clear_inputs
 - Bse::Source, 206
- clear_outputs
 - Bse::Source, 206
- clear_undo
 - Bse::Project, 149
- close
 - Bse::EditableSample, 88
- collect_runtime_entropy
 - Bse, 25
- collect_stats
 - Bse::EditableSample, 88
- collect_system_entropy
 - Bse, 25
- color
 - Bse::AnsiColors, 48
- color_code
 - Bse::AnsiColors, 48
- colorize_tty
 - Bse::AnsiColors, 48
- Colors
 - Bse::AnsiColors, 48
- common_ancestor
 - Bse::Item, 95
- config_dirs
 - Bse::Path, 51
- config_home
 - Bse::Path, 51
- config_names
 - Bse::Path, 51, 52
- configure
 - Bse::AnsiColors, 49
- connect_bus
 - Bse::Bus, 64
- connect_track
 - Bse::Bus, 64
- constexpr_equals
 - Bse, 25
- construct
 - Bse::FriendAllocator, 93
- ConvertAny, 74
- copy_reordered
 - Bse, 25
- cpu_arch
 - Bse, 26
- cpu_info
 - Bse, 26
- create
 - Bse::Flac1Handle, 91
- create_bus
 - Bse::Song, 195
- create_child
 - Bse::Xms::SerializationNode, 164
- create_csynth
 - Bse::Project, 149
- create_device
 - Bse::Track, 228
- create_memory_area
 - Bse, 26
- create_midi_synth
 - Bse::Project, 150
- create_module
 - Bse::Device, 81
- create_part
 - Bse::Song, 195
- create_project
 - Bse::Server, 171
- create_signal_monitor
 - Bse::Source, 207
- create_song
 - Bse::Project, 150
- create_source
 - Bse::SNet, 191
- create_track
 - Bse::Song, 195

- cstime
 - Bse::TaskStatus, 224
- cstrings_to_vector
 - Bse, 26
- current_locale_strtold
 - Bse, 26
- cutime
 - Bse::TaskStatus, 224
- cwd
 - Bse::Path, 52
- data
 - Bse::Blob, 61
- data_dirs
 - Bse::Path, 52
- data_home
 - Bse::Path, 52
- deactivate
 - Bse::Project, 150
- debug
 - Bse, 26
- debug_enabled
 - Bse, 26
- debug_key_enabled
 - Bse, 27
- debug_key_value
 - Bse, 27
- debug_message
 - Bse, 27
- debug_name
 - Bse::LegacyObject, 114
- delay
 - Bse::Resampler2, 157
- delete_data
 - Bse::DataListContainer, 79
- delete_event
 - Bse::Part, 133
- delete_inplace
 - Bse, 27
- deselect_controls
 - Bse::Part, 133
- deselect_event
 - Bse::Part, 133
- deselect_notes
 - Bse::Part, 133
- destroy
 - Bse::DataKey, 77
 - Bse::FriendAllocator, 93
- destroy_project
 - Bse::Server, 171
- device_type_info
 - Bse::Device, 81
 - Bse::Track, 228
- diag_abort_hook
 - Bse, 27
- digest
 - Bse::SHA3_224, 179
 - Bse::SHA3_256, 180
 - Bse::SHA3_384, 181
 - Bse::SHA3_512, 182
- dirname
 - Bse::Path, 52
- discard
 - Bse::KeccakRng, 110
- disconnect_bus
 - Bse::Bus, 64
- disconnect_track
 - Bse::Bus, 64
- editable_property
 - Bse::Item, 95
- engine_active
 - Bse::Server, 171
- ensure_master_bus
 - Bse::Song, 195
- ensure_output
 - Bse::Bus, 64
 - Bse::Track, 228
- ensure_track_links
 - Bse::Song, 196
- entity
 - Bse::AuxData, 60
- equals
 - Bse::Path, 52
- exec_handler_clear
 - Bse, 27
- exec_now
 - Bse, 27, 28
- exec_timeout
 - Bse, 28
- executable_name
 - Bse, 28
- executable_path
 - Bse, 28
- expand_tilde
 - Bse::Path, 52
- fallback
 - Bse::DataKey, 77
- fatal_error
 - Bse, 28
- feature_check
 - Bse, 28
- feature_toggle_bool
 - Bse, 29
- feature_toggle_find
 - Bse, 29
- find_any_track_for_part
 - Bse::Song, 196
- find_memory_area
 - Bse, 29
- find_module_type
 - Bse::Server, 171
- find_precision_for_bits
 - Bse::Resampler2, 157
- find_prop
 - Bse::Object, 128
- find_track_for_part
 - Bse::Song, 196
- find_typedata
 - Bse::LegacyObject, 114
- first_child
 - Bse::Xms::SerializationNode, 164
- fmsb
 - Bse, 29
- forget
 - Bse::KeccakRng, 110
- format
 - Bse::Lib::StringFormatter, 213
- from_file
 - Bse::Blob, 62
- from_proxy
 - Bse::Server, 171
- from_url
 - Bse::Blob, 62
- generate
 - Bse::AutoSeeder, 59
 - Bse::KeccakRng, 110
- get
 - Bse::Xms::SerializationNode, 164
- get_automation_channel
 - Bse::Source, 207
- get_automation_control
 - Bse::Source, 207
- get_channel_controls
 - Bse::Part, 134
- get_config
 - Bse::Server, 171
- get_config_defaults
 - Bse::Server, 171
- get_controls
 - Bse::Part, 134
- get_custom_effect_dir
 - Bse::Server, 172
- get_custom_instrument_dir
 - Bse::Server, 172
- get_data
 - Bse::DataListContainer, 79
- get_demo_path
 - Bse::Server, 172
- get_device_type
 - Bse::Device, 81
- get_effect_path
 - Bse::Server, 172
- get_frame_duration
 - Bse::SignalMonitor, 187
- get_instrument_path
 - Bse::Server, 172
- get_item
 - Bse::Container, 68
- get_ladspa_path
 - Bse::Server, 172
- get_last_tick
 - Bse::Part, 134
 - Bse::Track, 228
- get_length
 - Bse::EditableSample, 88
- get_master_bus
 - Bse::Song, 196
- get_max_note
 - Bse::Part, 134
- get_midi_notifier
 - Bse::Project, 150
- get_min_note
 - Bse::Part, 134
- get_mix_freq
 - Bse::SignalMonitor, 187
- get_source
 - Bse::Source, 207
- get_module_type
 - Bse::Module, 124
- get_mp3_version
 - Bse::Server, 172
- get_n_channels
 - Bse::EditableSample, 88

- get_name
 - Bse::Item, 96
- get_name_or_type
 - Bse::Item, 96
- get_notes
 - Bse::Part, 134
- get_ochannel
 - Bse::SignalMonitor, 187
- get_osc_freq
 - Bse::EditableSample, 88
- get_osource
 - Bse::SignalMonitor, 187
- get_output_source
 - Bse::Track, 229
- get_parent
 - Bse::Item, 96
- get_part
 - Bse::Track, 229
- get_plugin_path
 - Bse::Server, 172
- get_probe_features
 - Bse::SignalMonitor, 187
- get_project
 - Bse::Item, 96
- get_prop
 - Bse::Object, 128
- get_property_candidates
 - Bse::Item, 96
- get_sample_path
 - Bse::Server, 172
- get_seqid
 - Bse::Item, 96
- get_shared_memory
 - Bse::Server, 172
- get_shm_offset
 - Bse::SignalMonitor, 187
 - Bse::Song, 196
- get_sound_font_repo
 - Bse::Project, 150
- get_state
 - Bse::Project, 150
- get_supers
 - Bse::Project, 150
- get_timing
 - Bse::Part, 134
 - Bse::Song, 196
 - Bse::Track, 229
- get_type
 - Bse::Item, 96
- get_type_authors
 - Bse::Item, 96
- get_type_blurb
 - Bse::Item, 96
- get_type_license
 - Bse::Item, 97
- get_type_name
 - Bse::Item, 97
- get_underscore_path
 - Bse::Item, 97
- get_version
 - Bse::Server, 173
- get_vorbis_version
 - Bse::Server, 173
- get_wave_repo
 - Bse::Project, 150
- group_undo
 - Bse::Item, 97
- has
 - Bse::Xms::SerializationNode, 165
- has_output
 - Bse::Source, 207
- has_outputs
 - Bse::Source, 207
- hex
 - Bse::Xms::SerializationField, 162
- ichannel_blurb
 - Bse::Source, 207
- ichannel_get_n_joints
 - Bse::Source, 207
- ichannel_get_ochannel
 - Bse::Source, 207
- ichannel_get_osource
 - Bse::Source, 208
- ichannel_ident
 - Bse::Source, 208
- ichannel_label
 - Bse::Source, 208
- icon_from_pixstream
 - Bse, 29
- icon_sanitize
 - Bse, 29
- import_midi_file
 - Bse::Project, 150
- in_load
 - Bse::Xms::SerializationNode, 165
- in_save
 - Bse::Xms::SerializationNode, 165
- info
 - Bse, 29
- init
 - Bse::Test, 55
- init_async
 - Bse, 30
- init_glue_context
 - Bse, 30
- init_needed
 - Bse, 30
- init_server_instance
 - Bse, 30
- inject_midi_control
 - Bse::Project, 151
- insert_control
 - Bse::Part, 134
- insert_note
 - Bse::Part, 135
- insert_note_auto
 - Bse::Part, 135
- insert_part
 - Bse::Track, 229
- int16
 - Bse, 20
- int32
 - Bse, 20
- int64
 - Bse, 20
- int8
 - Bse, 20
- internal
 - Bse::Item, 97
- is_active
 - Bse::Project, 151
- is_dirty
 - Bse::Project, 151
- is_event_selected
 - Bse::Part, 135
- is_joint_ichannel
 - Bse::Source, 208
- is_joint_ichannel_by_id
 - Bse::Source, 208
- is_playing
 - Bse::Project, 151
- is_prepared
 - Bse::Source, 208
- isabs
 - Bse::Path, 52
- isdirname
 - Bse::Path, 53
- KAMMER_FREQ
 - Bse, 47
- KAMMER_NOTE
 - Bse, 47
- KAMMER_OCTAVE
 - Bse, 47
- KeccakCryptoRng
 - Bse::KeccakCryptoRng, 103
- KeccakF1600
 - Bse::Lib::KeccakF1600, 104
- KeccakFastRng
 - Bse::KeccakFastRng, 106
- KeccakGoodRng
 - Bse::KeccakGoodRng, 107
- KeccakRng
 - Bse::KeccakRng, 110
- label
 - Bse::UserMessage, 234
- last_project
 - Bse::Server, 173
- list
 - Bse::TaskRegistry, 222
- list_children
 - Bse::Container, 68
- list_controls
 - Bse::Part, 135
- list_device_types
 - Bse::Track, 229
- list_devices
 - Bse::Track, 229
- list_links
 - Bse::Part, 135
- list_midi_drivers
 - Bse::Server, 173
- list_module_types
 - Bse::Device, 81
 - Bse::Server, 173
- list_modules
 - Bse::Device, 81
- list_notes_crossing
 - Bse::Part, 135
- list_notes_within
 - Bse::Part, 135
- list_parts

- Bse::Track, 229
- list_parts_uniq
 - Bse::Track, 229
- list_pcm_drivers
 - Bse::Server, 173
- list_props
 - Bse::Object, 128
- list_selected_controls
 - Bse::Part, 136
- list_selected_notes
 - Bse::Part, 136
- list_tracks
 - Bse::Song, 196
- load
 - Bse::Xms::SerializationNode, 165
- load_assets
 - Bse::Server, 173
- load_file
 - Bse::WaveRepo, 244
- loading
 - Bse::Xms::SerializationNode, 165
- locked_config
 - Bse::Server, 173
- lookup_item
 - Bse::Container, 68

- MAX_OCTAVE
 - Bse, 47
- MIN_FINE_TUNE
 - Bse, 47
- MIN_OCTAVE
 - Bse, 47
- make_shared
 - Bse::FriendAllocator, 93
- max
 - Bse::KeccakRng, 110
- max_elapsed
 - Bse::Test::Timer, 226
- memset4
 - Bse, 30
- min
 - Bse::KeccakRng, 110
- min_elapsed
 - Bse::Test::Timer, 226
- mkdirs
 - Bse::Path, 53
- module_type_icon
 - Bse::Server, 173
- module_type_info
 - Bse::Device, 81
 - Bse::Module, 124
- ModuleFlag
 - Bse, 21
- MonitorField
 - Bse, 21
- monotonic_counter
 - Bse, 30

- n_channels
 - Bse::Source, 208
- n_nums
 - Bse::KeccakRng, 111
- n_ochannels
 - Bse::Source, 208
- n_reps
 - Bse::Test::Timer, 226
- n_wave_chunks
 - Bse::Wave, 236
- name
 - Bse::Blob, 62
 - Bse::TaskStatus, 224
 - Bse::Xms::SerializationNode, 165
- new_inplace
 - Bse, 30
- node
 - Bse::Xms::SerializationField, 162
- note_construct
 - Bse::Server, 173
- note_describe
 - Bse::Server, 174
- note_from_freq
 - Bse::Server, 174
- note_from_string
 - Bse::Server, 174
- note_to_freq
 - Bse::Server, 174
- notify
 - Bse::Object, 128
- null_id
 - Bse::Xms::SerializationNode, 166

- ochannel_blurb
 - Bse::Source, 208
- ochannel_ident
 - Bse::Source, 209
- ochannel_label
 - Bse::Source, 209
- open
 - Bse::EditableSample, 89
- operator &
 - Bse::Xms::SerializationField, 162
- operator bool
 - Bse::Blob, 62
 - Bse::Xms::SerializationNode, 165
- operator! =
 - Bse::KeccakRng, 111
- operator <<
 - Bse::KeccakRng, 112
- operator >>
 - Bse::KeccakRng, 112
- operator()
 - Bse::AutoSeeder, 59
 - Bse::KeccakRng, 111
- operator = =
 - Bse::KeccakRng, 112
- operator[]
 - Bse::Xms::SerializationNode, 165
- order
 - Bse::Resampler2, 157
- outputs
 - Bse::Track, 230

- parse_xml
 - Bse::Xms::SerializationNode, 165
- Pcg32Rng
 - Bse::Pcg32Rng, 141, 142
- permute
 - Bse::Lib::KeccakF1600, 104
- play
 - Bse::Project, 151

- posix_locale
 - Bse::Lib::ScopedPosixLocale, 160
- posix_locale_strtold
 - Bse, 31
- precision_name
 - Bse::Resampler2, 157
- print_backtrace
 - Bse, 31
- printerr
 - Bse, 31
- printout
 - Bse, 31
- priority
 - Bse::TaskStatus, 224
- process_block
 - Bse::Resampler2, 158
- process_id
 - Bse::TaskStatus, 224
- processor
 - Bse::TaskStatus, 224
- program_alias
 - Bse, 31
- program_alias_init
 - Bse, 31
- program_cwd
 - Bse, 31
- ProjectState
 - Bse, 22
- proxy_id
 - Bse::LegacyObject, 114
- purge_stale_cachedirs
 - Bse::Server, 174
- push_property_undo
 - Bse::ItemImpl, 101
- push_undo
 - Bse::ItemImpl, 101
- push_undo_to_redo
 - Bse::ItemImpl, 102
- put_wstore
 - Bse::Flac1Handle, 91

- queue_controls
 - Bse::Part, 136
- queue_notes
 - Bse::Part, 136

- random
 - Bse::AutoSeeder, 59
 - Bse::KeccakRng, 111
 - Bse::Pcg32Rng, 142
- random_float
 - Bse, 32
 - Bse::Test, 55
- random_frange
 - Bse, 32
 - Bse::Test, 55
- random_int64
 - Bse, 32
 - Bse::Test, 55
- random_irange
 - Bse, 32
 - Bse::Test, 55
- random_nonce
 - Bse, 32
- random_secret

- Bse, 32
- read_data
 - Bse::Flac1Handle, 91
- realpath
 - Bse::Path, 53
- redo
 - Bse::Project, 151
- redo_depth
 - Bse::Project, 151
- remlink
 - Bse::Xms::SerializationNode, 166
- register_core_plugins
 - Bse::Server, 174
- register_ladspa_plugins
 - Bse::Server, 174
- register_source_module
 - Bse::ServerImpl, 178
- release_aligned_block
 - Bse, 32
- remove
 - Bse::TaskRegistry, 222
- remove_bus
 - Bse::Song, 196
- remove_link
 - Bse::Track, 229
- remove_part
 - Bse::Song, 196
- remove_snet
 - Bse::Project, 151
- remove_source
 - Bse::SNet, 191
- remove_tick
 - Bse::Track, 230
- remove_track
 - Bse::Song, 197
- remove_wave
 - Bse::WaveRepo, 244
- request_pcm_position
 - Bse::WaveOsc, 240
- Resampler2
 - Bse::Resampler2, 157
- reset
 - Bse::Lib::KeccakF1600, 105
 - Bse::Resampler2, 158
 - Bse::SHA3_224, 179
 - Bse::SHA3_256, 180
 - Bse::SHA3_384, 181
 - Bse::SHA3_512, 182
 - Bse::SHAKE128, 183
 - Bse::SHAKE256, 184
- restore_from_file
 - Bse::Project, 151
- result_type
 - Bse::KeccakRng, 109
- run
 - Bse::Test, 55, 56
- runpath
 - Bse, 32
- runtime_dir
 - Bse::Path, 53
- SHA3_224
 - Bse::SHA3_224, 178
- SHA3_256
 - Bse::SHA3_256, 179
- SHA3_384
 - Bse::SHA3_384, 180
- SHA3_512
 - Bse::SHA3_512, 181
- SHAKE128
 - Bse::SHAKE128, 182
- SHAKE256
 - Bse::SHAKE256, 183
- sample_file_info
 - Bse::Server, 174
- save
 - Bse::Xms::SerializationNode, 166
- save_under
 - Bse::Xms::SerializationNode, 166
- searchpath_contains
 - Bse::Path, 53
- searchpath_find
 - Bse::Path, 53
- searchpath_list
 - Bse::Path, 53
- searchpath_multiply
 - Bse::Path, 53
- seed
 - Bse::KeccakRng, 111
 - Bse::Pcg32Rng, 142
- select_controls
 - Bse::Part, 136
- select_controls_exclusive
 - Bse::Part, 136
- select_event
 - Bse::Part, 136
- select_notes
 - Bse::Part, 137
- select_notes_exclusive
 - Bse::Part, 137
- send_user_message
 - Bse::Server, 175
- serialization_node
 - Bse::Xms::SerializationField, 162
- SerializationNode
 - Bse::Xms::SerializationNode, 164
- set_automation
 - Bse::Source, 209
- set_config
 - Bse::Server, 175
- set_data
 - Bse::DataListContainer, 79
- set_from_editable_sample
 - Bse::WaveOsc, 240
- set_input
 - Bse::Source, 209
- set_input_by_id
 - Bse::Source, 209
- set_name
 - Bse::Item, 97
- set_pos
 - Bse::Source, 209
- set_probe_features
 - Bse::SignalMonitor, 187
- set_prop
 - Bse::Object, 128
- Sfi, 56
- SfiRecFields, 178
- sha3_224_hash
 - Bse, 33
- sha3_256_hash
 - Bse, 33
- sha3_384_hash
 - Bse, 33
- sha3_512_hash
 - Bse, 33
- shake128_hash
 - Bse, 33
- shake256_hash
 - Bse, 33
- shared_ptr_cast
 - Bse, 34
- shm_creator
 - Bse::SharedMemory, 184
- shm_length
 - Bse::SharedMemory, 185
- shm_start
 - Bse::SharedMemory, 185
- size
 - Bse::Blob, 62
- slow
 - Bse::Test, 56
- SongTelemetry
 - Bse, 22
- squeeze_digest
 - Bse::SHAKE128, 183
 - Bse::SHAKE256, 184
- sse_available
 - Bse::Resampler2, 158
- sse_enabled
 - Bse::Resampler2, 158
- start_playback
 - Bse::Project, 152
- start_recording
 - Bse::Server, 175
- State
 - Bse::Project, 152
- state
 - Bse::TaskStatus, 224
- stime
 - Bse::TaskStatus, 225
- stop
 - Bse::Project, 152
- stop_playback
 - Bse::Project, 152
- store
 - Bse::Project, 152
- store_bse
 - Bse::Project, 152
- String
 - Bse, 20
- string
 - Bse::Blob, 62
 - Bse::TaskStatus, 224
- string_canonify
 - Bse, 34
- string_capitalize
 - Bse, 35
- string_casecmp
 - Bse, 35
- string_casefold
 - Bse, 35
- string_cmp
 - Bse, 35
- string_cmp_uuid

- Bse, 35
- string_endswith
 - Bse, 35
- string_format
 - Bse, 35
- string_from_bool
 - Bse, 35
- string_from_cquote
 - Bse, 36
- string_from_double
 - Bse, 36
- string_from_double_vector
 - Bse, 36
- string_from_errno
 - Bse, 36
- string_from_float
 - Bse, 36
- string_from_int
 - Bse, 36
- string_from_pretty_function_name
 - Bse, 36
- string_from_type
 - Bse, 36
- string_from_uint
 - Bse, 37
- string_from_unicode
 - Bse, 37
- string_has_int
 - Bse, 37
- string_hexdump
 - Bse, 37
- string_is_canonified
 - Bse, 37
- string_is_uuid
 - Bse, 37
- string_join
 - Bse, 37
- string_locale_format
 - Bse, 38
- string_locale_vprintf
 - Bse, 38
- string_lstrip
 - Bse, 38
- string_match_identifier
 - Bse, 38
- string_match_identifier_tail
 - Bse, 38
- string_multiply
 - Bse, 38
- string_normalize_nfc
 - Bse, 38
- string_normalize_nfd
 - Bse, 39
- string_normalize_nfkc
 - Bse, 39
- string_normalize_nfkd
 - Bse, 39
- string_option_check
 - Bse, 39
- string_option_get
 - Bse, 39
- string_options_split
 - Bse, 39
- string_replace
 - Bse, 39
- string_rstrip
 - Bse, 40
- string_set_A2Z
 - Bse, 40
- string_set_a2z
 - Bse, 40
- string_set_ascii_alnum
 - Bse, 40
- string_split
 - Bse, 40
- string_split_any
 - Bse, 40
- string_startswith
 - Bse, 40
- string_strip
 - Bse, 40
- string_substitute_char
 - Bse, 40
- string_to_bool
 - Bse, 41
- string_to_cescape
 - Bse, 41
- string_to_cquote
 - Bse, 41
- string_to_double
 - Bse, 41
- string_to_double_vector
 - Bse, 41
- string_to_int
 - Bse, 41
- string_to_type
 - Bse, 42
- string_to_uint
 - Bse, 42
- string_tolower
 - Bse, 42
- string_totitle
 - Bse, 42
- string_toupper
 - Bse, 42
- string_vector_erase_empty
 - Bse, 42
- string_vector_find
 - Bse, 42
- string_vector_find_value
 - Bse, 43
- string_vector_lstrip
 - Bse, 43
- string_vector_rstrip
 - Bse, 43
- string_vector_strip
 - Bse, 43
- string_vprintf
 - Bse, 43
- StringVector
 - Bse, 20
- stringify_arg
 - Bse::Test, 56
- supports_user_synths
 - Bse::SNet, 191
- swap_data
 - Bse::DataListContainer, 79
- sync_seek_perc
 - Bse::WaveOsc, 240
- synthesize_note
 - Bse::Song, 197
- task_id
 - Bse::TaskStatus, 225
- TaskStatus
 - Bse::TaskStatus, 223
- test_counter_inc_fetch
 - Bse::Server, 175
- test_elapsed
 - Bse::Test::Timer, 226
- test_filter_impl
 - Bse::Resampler2, 158
- text1
 - Bse::UserMessage, 234
- text2
 - Bse::UserMessage, 234
- text3
 - Bse::UserMessage, 234
- text_convert
 - Bse, 43
- tick_stamp_from_systime
 - Bse::Server, 175
- Timer
 - Bse::Test::Timer, 225
- timestamp_benchmark
 - Bse, 44
- timestamp_format
 - Bse, 44
- timestamp_realtime
 - Bse, 44
- timestamp_resolution
 - Bse, 44
- timestamp_startup
 - Bse, 44
- title
 - Bse::UserMessage, 234
- uint
 - Bse, 21
- uint16
 - Bse, 21
- uint32
 - Bse, 21
- uint64
 - Bse, 21
- uint8
 - Bse, 21
- undo
 - Bse::Project, 152
- undo_depth
 - Bse::Project, 152
- undo_descriptor
 - Bse::ItemImpl, 102
- undo_resolve
 - Bse::ItemImpl, 102
- ungroup_undo
 - Bse::Item, 97
- unichar
 - Bse, 21
- unicode_is_assigned
 - Bse, 44
- unicode_is_character
 - Bse, 44
- unicode_is_control_code
 - Bse, 44

- unicode_is_noncharacter
 - Bse, 44
- unicode_is_private
 - Bse, 45
- unicode_is_valid
 - Bse, 45
- unique_id
 - Bse::LegacyObject, 114
- unset_input
 - Bse::Source, 210
- unset_input_by_id
 - Bse::Source, 210
- unuse
 - Bse::Item, 97
- update
 - Bse::SHA3_224, 179
 - Bse::SHA3_256, 180
 - Bse::SHA3_384, 181
 - Bse::SHA3_512, 182
 - Bse::SHAKE128, 183
 - Bse::SHAKE256, 184
- Bse::TaskRegistry, 222
- Bse::TaskStatus, 224
- url_show
 - Bse, 45
- use
 - Bse::Item, 97
- use_editable
 - Bse::Wave, 236
- user_home
 - Bse::Path, 54
- UserMessageType
 - Bse, 22
- utf8_to_unicode
 - Bse, 45
- utf8len
 - Bse, 45
- utime
 - Bse::TaskStatus, 225
- utype
 - Bse::UserMessage, 234
- vector_erase_element
 - Bse, 46
- vector_erase_iface
 - Bse, 46
- verbose
 - Bse::Test, 56
- version
 - Bse, 46
- warning
 - Bse, 46
- with_default
 - Bse::Xms::SerializationNode, 166
- write_xml
 - Bse::Xms::SerializationNode, 166
- xor_seed
 - Bse::KeccakRng, 111
- zintern_decompress
 - Bse, 46
- zintern_free
 - Bse, 47