QLaibLibCpp

0.1.0

Generated by Doxygen 1.9.8

1 QLaibLib Architecture (C++/Qt) 1
1.1 Layout (cpp/) 1
1.2 Dataflow 1
1.3 Build dependencies 1
1.4 Runtime 2

2 QLaibLib Usage (C++ Qt GUI) 3
2.1 Run modes 3
2.2 Controls 3
2.2.1 Pairs 3
2.2.2 Histogram 3
2.2.3 Plots 4
2.2.4 Export 4
2.3 Env vars 4
2.4 Windows notes 4

3 Namespace Index 5
3.1 Namespace List 5

4 Hierarchical Index 7
4.1 Class Hierarchy 7

5 Class Index 9
5.1 Class List 9

6 File Index 11
6.1 File List 11

7 Namespace Documentation 13
7.1 qlaib Namespace Reference 13
7.2 qlaib::acquisition Namespace Reference 13
7.3 qlaib::acquisition::anonymous_namespace{QuTAGBackend.cpp} Namespace Reference 13
7.3.1 Variable Documentation 14
7.3.1.1 kDefaultChannelMask 14
7.4 qlaib::core Namespace Reference 14
7.5 qlaib::data Namespace Reference 14
7.6 qlaib::metrics Namespace Reference 14
7.7 qlaib::ui Namespace Reference 14

8 Class Documentation 15
8.1 qlaib::acquisition::BackendConfig Struct Reference 15
8.1.1 Detailed Description 15
8.1.2 Member Data Documentation 15
8.1.2.1 coincidenceWindowPs 15
8.1.2.2 exposureSeconds 16

8.1.2.3 replayFile . . . 16
8.1.2.4 timestampBufferSize . . . 16
8.1.2.5 useMock . . . 16
8.2 qlaib::acquisition::BinReplayBackend Class Reference . . . 16
8.2.1 Member Function Documentation . . . 17
8.2.1.1 nextBatch() . . . 17
8.2.1.2 start() . . . 18
8.2.1.3 stop() . . . 19
8.2.2 Member Data Documentation . . . 19
8.2.2.1 coincWindowPs_ . . . 19
8.2.2.2 currentSecond_ . . . 19
8.2.2.3 lastSecond_ . . . 19
8.2.2.4 singles_ . . . 19
8.3 qlaib::data::Coincidence Struct Reference . . . 19
8.3.1 Member Data Documentation . . . 20
8.3.1.1 counts . . . 20
8.3.1.2 label . . . 20
8.4 qlaib::metrics::DummyQBER Class Reference . . . 20
8.4.1 Member Function Documentation . . . 21
8.4.1.1 compute() . . . 21
8.4.1.2 name() . . . 21
8.5 qlaib::metrics::DummyVisibility Class Reference . . . 21
8.5.1 Member Function Documentation . . . 22
8.5.1.1 compute() . . . 22
8.5.1.2 name() . . . 22
8.6 qlaib::core::EventBus< Payload > Class Template Reference . . . 22
8.6.1 Member Typedef Documentation . . . 23
8.6.1.1 Callback . . . 23
8.6.2 Member Function Documentation . . . 23
8.6.2.1 publish() . . . 23
8.6.2.2 subscribe() . . . 23
8.6.3 Member Data Documentation . . . 23
8.6.3.1 mutex_ . . . 23
8.6.3.2 nextToken_ . . . 24
8.6.3.3 subscribers_ . . . 24
8.7 qlaib::acquisition::IBackend Class Reference . . . 24
8.7.1 Constructor & Destructor Documentation . . . 25
8.7.1.1 ~IBackend() . . . 25
8.7.2 Member Function Documentation . . . 25
8.7.2.1 nextBatch() . . . 25
8.7.2.2 start() . . . 25
8.7.2.3 stop() . . . 25

8.8 qlaib::metrics::IMetric Class Reference . . . 26
8.8.1 Constructor & Destructor Documentation . . . 26
8.8.1.1 ~IMetric() . . . 26
8.8.2 Member Function Documentation . . . 26
8.8.2.1 compute() . . . 26
8.8.2.2 name() . . . 26
8.9 qlaib::ui::MainWindow Class Reference . . . 27
8.9.1 Detailed Description . . . 28
8.9.2 Constructor & Destructor Documentation . . . 28
8.9.2.1 MainWindow() . . . 28
8.9.3 Member Function Documentation . . . 28
8.9.3.1 addPair . . . 28
8.9.3.2 appendSample() . . . 29
8.9.3.3 calibrateAll . . . 29
8.9.3.4 calibrateSelected . . . 29
8.9.3.5 computeHistogram . . . 29
8.9.3.6 exportCsv . . . 29
8.9.3.7 loadConfig() . . . 29
8.9.3.8 refreshPairList() . . . 29
8.9.3.9 refreshPairsTable() . . . 29
8.9.3.10 removeSelectedPair . . . 29
8.9.3.11 resetPairs . . . 29
8.9.3.12 saveConfig() . . . 30
8.9.3.13 setMode() . . . 30
8.9.3.14 setReplayFile() . . . 30
8.9.3.15 setupUi() . . . 30
8.9.3.16 start() . . . 30
8.9.3.17 stop() . . . 30
8.9.3.18 tick . . . 30
8.9.3.19 toggleRecording . . . 30
8.9.4 Member Data Documentation . . . 30
8.9.4.1 backend_ . . . 30
8.9.4.2 cfg_ . . . 31
8.9.4.3 coincidenceWindowPs_ . . . 31
8.9.4.4 histogramWindowPs_ . . . 31
8.9.4.5 latestBatch_ . . . 31
8.9.4.6 metrics_ . . . 31
8.9.4.7 mode_ . . . 31
8.9.4.8 pairs_ . . . 31
8.9.4.9 replayFile_ . . . 31
8.9.4.10 t0_ms_ . . . 31
8.9.4.11 timer_ . . . 31

8.10 qlaib::acquisition::MockBackend Class Reference . . . 32
8.10.1 Member Function Documentation . . . 33
8.10.1.1 nextBatch() . . . 33
8.10.1.2 start() . . . 33
8.10.1.3 stop() . . . 33
8.10.2 Member Data Documentation . . . 34
8.10.2.1 exposure_ . . . 34
8.10.2.2 rng_ . . . 34
8.10.2.3 running_ . . . 34
8.10.2.4 startTime_ . . . 34
8.11 qlaib::ui::PairSpec Struct Reference . . . 34
8.11.1 Detailed Description . . . 34
8.11.2 Member Data Documentation . . . 35
8.11.2.1 chA . . . 35
8.11.2.2 chB . . . 35
8.11.2.3 delayPs . . . 35
8.11.2.4 label . . . 35
8.12 qlaib::acquisition::QuTAGBackend Class Reference . . . 35
8.12.1 Member Function Documentation . . . 36
8.12.1.1 nextBatch() . . . 36
8.12.1.2 start() . . . 37
8.12.1.3 startRecording() . . . 37
8.12.1.4 stop() . . . 37
8.12.1.5 stopRecording() . . . 37
8.12.2 Member Data Documentation . . . 37
8.12.2.1 bufferSize_ . . . 37
8.12.2.2 chBuffer_ . . . 37
8.12.2.3 coincWindowPs_ . . . 38
8.12.2.4 connected_ . . . 38
8.12.2.5 exposureMs_ . . . 38
8.12.2.6 recording_ . . . 38
8.12.2.7 tsBuffer_ . . . 38
8.13 qlaib::metrics::Registry Class Reference . . . 38
8.13.1 Member Function Documentation . . . 38
8.13.1.1 computeAll() . . . 38
8.13.1.2 registerMetric() . . . 39
8.13.2 Member Data Documentation . . . 39
8.13.2.1 metrics_ . . . 39
8.14 qlaib::data::SampleBatch Struct Reference . . . 39
8.14.1 Member Data Documentation . . . 39
8.14.1.1 coincidences . . . 39
8.14.1.2 metrics . . . 39

8.14.1.3 singles . . . 39
8.14.1.4 timestamp . . . 40
8.14.1.5 timestamps_ps . . . 40
8.15 qlaib::core::EventBus< Payload >::Subscriber Struct Reference . . . 40
8.15.1 Member Data Documentation . . . 40
8.15.1.1 cb . . . 40
8.15.1.2 token . . . 40

9 File Documentation **41**
9.1 include/qlaib/acquisition/BinReplayBackend.h File Reference . . . 41
9.2 BinReplayBackend.h . . . 42
9.3 include/qlaib/acquisition/IBackend.h File Reference . . . 42
9.4 IBackend.h . . . 43
9.5 include/qlaib/acquisition/MockBackend.h File Reference . . . 44
9.6 MockBackend.h . . . 45
9.7 include/qlaib/acquisition/QuTAGBackend.h File Reference . . . 45
9.8 QuTAGBackend.h . . . 46
9.9 include/qlaib/core/EventBus.h File Reference . . . 47
9.10 EventBus.h . . . 48
9.11 include/qlaib/data/SampleBatch.h File Reference . . . 48
9.12 SampleBatch.h . . . 49
9.13 include/qlaib/metrics/IMetric.h File Reference . . . 50
9.14 IMetric.h . . . 51
9.15 include/qlaib/metrics/Registry.h File Reference . . . 51
9.16 Registry.h . . . 52
9.17 include/qlaib/ui/MainWindow.h File Reference . . . 53
9.18 MainWindow.h . . . 54
9.19 src/acquisition/BinReplayBackend.cpp File Reference . . . 55
9.20 src/acquisition/MockBackend.cpp File Reference . . . 56
9.21 src/acquisition/QuTAGBackend.cpp File Reference . . . 56
9.22 src/metrics/Registry.cpp File Reference . . . 57
9.23 src/ui/MainWindow.cpp File Reference . . . 57
9.23.1 Function Documentation . . . 58
9.23.1.1 findBestDelayPicoseconds() . . . 58
9.24 /home/bogfootlj/Documents/PhDCode/QLaibLibCpp/docs/ARCHITECTURE.md File Reference . . . 59
9.25 /home/bogfootlj/Documents/PhDCode/QLaibLibCpp/docs/USAGE.md File Reference . . . 59

Chapter 1

QLaibLib Architecture (C++/Qt)

This snapshot focuses on the native C++/Qt GUI driven by coincfinder and the quTAG SDK. The Python package is intentionally omitted/ignored.

1.1 Layout (cpp/)

```
cpp/
  include/qlaib/
       acquisition/      # IBackend + Mock, BinReplay, QuTAG backends
       data/             # SampleBatch, coincidence structs
       metrics/          # Metric interfaces and registry
       ui/               # Qt MainWindow
  src/                    # backend + UI implementations
  apps/
       qlaib_gui.cpp          # Qt GUI entry (modes: live/replay/mock)
       qlaib_record_qudag.cpp # headless BIN recorder
  CMakeLists.txt               # Qt6, coincfinder, optional quTAG
```

1.2 Dataflow

- **Backends**
  - `MockBackend` — synthetic 8-channel singles/timestamps.
  - `BinReplayBackend` — reads BIN → per-second singles/timestamps.
  - `QuTAGBackend` — polls `TDC_getLastTimestamps`, optional BIN recording via `TDC_write↩Timestamps`.
- **Coincidences**: computed in UI with coincfinder (`countCoincidencesWithDelay`, `findBest↩DelayPicoseconds`).
- **GUI**: Qt Charts tabs (Singles, Coincidences, Metrics, Histogram); configurable pair table; auto-delay calibration; CSV export; BIN record (quTAG only); settings persisted to JSON config.

1.3 Build dependencies

- Qt6 Core/Widgets/Charts
- C++20 compiler
- coincfinder sources (compiled directly if static lib absent)
- quTAG SDK: `libtdcbase` (Linux) or `tdcbase.dll/.lib` plus deps (Windows), gated by `QQL_↩ENABLE_QUTAG`.

1.4 Runtime

- Headless/offscreen via `QLAIB_HEADLESS=1` (sets `QT_QPA_PLATFORM=offscreen`).
- Backend selection via CLI: `--mode=live|replay|mock` (live default), `--replay-bin <file>` for replay.

Chapter 2

QLaibLib Usage (C++ Qt GUI)

2.1 Run modes

- Live (default): `./cpp/build/apps/qlaib_gui` (tries quTAG, falls back to mock)
- BIN replay: `./cpp/build/apps/qlaib_gui --mode=replay --replay-bin /path/to/file.bin`
- Mock: `./cpp/build/apps/qlaib_gui --mode=mock`
- Headless: add `QLAIB_HEADLESS=1` (forces Qt offscreen if no display).
- Record BIN (quTAG): `./cpp/build/apps/qlaib_record_qudag out.bin 1000` or the "↩ Record BIN" button.

2.2 Controls

- **Start/Stop:** begin/stop acquisition.
- **Exposure (s):** sets quTAG exposure / mock cadence.
- **Coinc window (ps):** coincidence window used for all pairs.

2.2.1 Pairs

- Table columns: Label, chA, chB, Delay (ps).
- Buttons: Add, Remove selected, Reset defaults, Calibrate selected/all.
- Edits apply immediately and persist to `~/.config/.../qlaib_gui.json`.

2.2.2 Histogram

- Select a pair, set Window/Start/End/Step (ps), click "Histogram" (switches to tab).
- Uses coincfinder `computeCoincidencesForRange` on latest batch timestamps.

2.2.3 Plots

- Tabs: Singles, Coincidences, Metrics, Histogram.
- Time axes grow from t=0; traces are not pruned.
- Legends update when pairs change; stale series are removed.

2.2.4 Export

- "Export CSV" writes all visible time-series (singles, coincidences, metrics).
- Settings (pairs, coincidence window) persist between runs.

2.3 Env vars

- `QLAIB_HEADLESS=1` – offscreen Qt platform (no X/Wayland).

2.4 Windows notes

- Install Qt6 + Charts (e.g., vcpkg `qtbase qtcharts`) in your own vcpkg clone (not the embedded VS copy).
- Build coincfinder: `cmake -S coincfinder -B coincfinder/build && cmake --build coincfinder/build`.
- Configure with `-DCOINCFINDER_CORE=path\to\coincfinder_core.lib -DTDCBASE_LIB=path\to\tdcbase.lib`.
- QuTAG SDK: use `DLL_64bit/tdcbase.lib` for linking; copy `DLL_64bit/tdcbase.dll` and its deps (`libusb0.dll`, `libgcc_s_seh-1.dll`, `libstdc++-6.dll`, `libwinpthread-1.dll`) into the run folder or add that directory to `PATH` before launching the GUI.
- If vcpkg is manifest-only, create a manifest in repo root: `vcpkg new --application`, `vcpkg add port qtbase`, `vcpkg add port qtcharts`, then `vcpkg install --triplet x64-windows`. Otherwise run installs from your vcpkg clone after `bootstrap-vcpkg.bat`.
- PowerShell use with a vcpkg clone at `C:\Users\you\vcpkg`:
```
$env:VCPKG_ROOT = "C:\Users\you\vcpkg"
& "$env:VCPKG_ROOT\vcpkg.exe" install qtbase qtcharts --triplet x64-windows
```
- vcpkg may pull ∼20–30 dependencies for Qt. To speed up, enable binary caching (`set VCPKG_FEATURE_FLAGS=manifests,binarycaching`) or install Qt via the official Qt online installer (select Qt 6.x MSVC 64-bit + Charts) and set `-DCMAKE_PREFIX_PATH="C:/Qt/6.x/msvc2019_64/lib/cmake"` in CMake. With vcpkg, point CMake at `installed\x64-windows\share\Qt6` or set `Qt6_DIR=...\share\Qt6\cmake`; ensure a space between CMake args on Windows.

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

qlaib 13
qlaib::acquisition 13
qlaib::acquisition::anonymous_namespace{QuTAGBackend.cpp} 13
qlaib::core 14
qlaib::data 14
qlaib::metrics 14
qlaib::ui 14

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

qlaib::acquisition::BackendConfig 15
qlaib::data::Coincidence 19
qlaib::core::EventBus< Payload > 22
qlaib::acquisition::IBackend 24
    qlaib::acquisition::BinReplayBackend 16
    qlaib::acquisition::MockBackend 32
    qlaib::acquisition::QuTAGBackend 35
qlaib::metrics::IMetric 26
    qlaib::metrics::DummyQBER 20
    qlaib::metrics::DummyVisibility 21
qlaib::ui::PairSpec 34
QMainWindow
    qlaib::ui::MainWindow 27
qlaib::metrics::Registry 38
qlaib::data::SampleBatch 39
qlaib::core::EventBus< Payload >::Subscriber 40

Chapter 5

Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

qlaib::acquisition::BackendConfig
: Immutable configuration shared by all acquisition backends . . . 15
qlaib::acquisition::BinReplayBackend . . . 16
qlaib::data::Coincidence . . . 19
qlaib::metrics::DummyQBER . . . 20
qlaib::metrics::DummyVisibility . . . 21
qlaib::core::EventBus< Payload > . . . 22
qlaib::acquisition::IBackend . . . 24
qlaib::metrics::IMetric . . . 26
qlaib::ui::MainWindow
: Main GUI window: drives backends and renders live metrics/plots . . . 27
qlaib::acquisition::MockBackend . . . 32
qlaib::ui::PairSpec
: User-configurable coincidence pair (channel A/B and relative delay) . . . 34
qlaib::acquisition::QuTAGBackend . . . 35
qlaib::metrics::Registry . . . 38
qlaib::data::SampleBatch . . . 39
qlaib::core::EventBus< Payload >::Subscriber . . . 40

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

include/qlaib/acquisition/BinReplayBackend.h . . . 41
include/qlaib/acquisition/IBackend.h . . . 42
include/qlaib/acquisition/MockBackend.h . . . 44
include/qlaib/acquisition/QuTAGBackend.h . . . 45
include/qlaib/core/EventBus.h . . . 47
include/qlaib/data/SampleBatch.h . . . 48
include/qlaib/metrics/IMetric.h . . . 50
include/qlaib/metrics/Registry.h . . . 51
include/qlaib/ui/MainWindow.h . . . 53
src/acquisition/BinReplayBackend.cpp . . . 55
src/acquisition/MockBackend.cpp . . . 56
src/acquisition/QuTAGBackend.cpp . . . 56
src/metrics/Registry.cpp . . . 57
src/ui/MainWindow.cpp . . . 57

Chapter 7

Namespace Documentation

7.1 qlaib Namespace Reference

Namespaces

- namespace acquisition
- namespace core
- namespace data
- namespace metrics
- namespace ui

7.2 qlaib::acquisition Namespace Reference

Namespaces

- namespace anonymous_namespace{QuTAGBackend.cpp}

Classes

- struct BackendConfig

  Immutable configuration shared by all acquisition backends.
- class BinReplayBackend
- class IBackend
- class MockBackend
- class QuTAGBackend

7.3 qlaib::acquisition::anonymous_namespace{QuTAGBackend.cpp} Namespace Reference

Variables

- constexpr int kDefaultChannelMask = 0xFF

7.3.1 Variable Documentation

7.3.1.1 kDefaultChannelMask

```
constexpr int qlaib::acquisition::anonymous_namespace{QuTAGBackend.cpp}::kDefaultChannelMask =
0xFF  [constexpr]
```

7.4 qlaib::core Namespace Reference

Classes

- class EventBus

7.5 qlaib::data Namespace Reference

Classes

- struct Coincidence
- struct SampleBatch

7.6 qlaib::metrics Namespace Reference

Classes

- class DummyQBER
- class DummyVisibility
- class IMetric
- class Registry

7.7 qlaib::ui Namespace Reference

Classes

- class MainWindow

  Main GUI window: drives backends and renders live metrics/plots.
- struct PairSpec

  User-configurable coincidence pair (channel A/B and relative delay).

Chapter 8

Class Documentation

8.1 qlaib::acquisition::BackendConfig Struct Reference

Immutable configuration shared by all acquisition backends.

`#include <IBackend.h>`

Public Attributes

- long long coincidenceWindowPs {200000}
  Coincidence window in picoseconds for coincidence counting.
- double exposureSeconds {1.0}
  Integration time used by backends that collect a batch per exposure.
- std::string replayFile
  BIN file path used when replaying recorded data.
- int timestampBufferSize {200000}
  Maximum timestamp buffer length for streaming (live quTAG).
- bool useMock {true}
  Whether the frontend explicitly requested the mock backend.

8.1.1 Detailed Description

Immutable configuration shared by all acquisition backends.

The GUI populates this from CLI flags / UI controls and passes it once into IBackend::start(). Backends should treat the struct as read-only after that point.

8.1.2 Member Data Documentation

8.1.2.1 coincidenceWindowPs

`long long qlaib::acquisition::BackendConfig::coincidenceWindowPs {200000}`

Coincidence window in picoseconds for coincidence counting.

8.1.2.2 exposureSeconds

```
double qlaib::acquisition::BackendConfig::exposureSeconds {1.0}
```

Integration time used by backends that collect a batch per exposure.

8.1.2.3 replayFile

```
std::string qlaib::acquisition::BackendConfig::replayFile
```

BIN file path used when replaying recorded data.

8.1.2.4 timestampBufferSize

```
int qlaib::acquisition::BackendConfig::timestampBufferSize {200000}
```

Maximum timestamp buffer length for streaming (live quTAG).

8.1.2.5 useMock

```
bool qlaib::acquisition::BackendConfig::useMock {true}
```

Whether the frontend explicitly requested the mock backend.

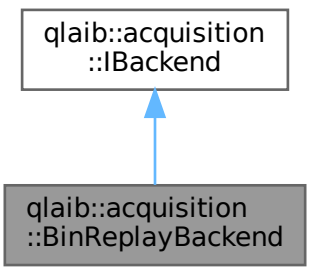
The documentation for this struct was generated from the following file:

- include/qlaib/acquisition/IBackend.h

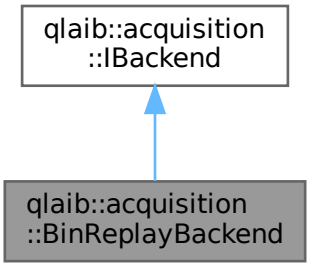
8.2 qlaib::acquisition::BinReplayBackend Class Reference

```
#include <BinReplayBackend.h>
```

Inheritance diagram for qlaib::acquisition::BinReplayBackend:



Collaboration diagram for qlaib::acquisition::BinReplayBackend:



Public Member Functions

- std::optional< data::SampleBatch > nextBatch () override

  Retrieve the next batch of samples, if available.
- bool start (const BackendConfig &config) override

  Initialise the backend and prepare to emit batches.
- void stop () override

  Stop acquisition and release any hardware or file handles.

Public Member Functions inherited from qlaib::acquisition::IBackend

- virtual ~IBackend ()=default

Private Attributes

- long long coincWindowPs_ {200000}
- long long currentSecond_ {0}
- long long lastSecond_ {-1}
- std::map< int, Singles > singles_

8.2.1 Member Function Documentation

8.2.1.1 nextBatch()

```
std::optional< data::SampleBatch > qlaib::acquisition::BinReplayBackend::nextBatch ( )  [override], [virtual]
```

Retrieve the next batch of samples, if available.

Returns

A SampleBatch when data is ready; std::nullopt otherwise.

Implements qlaib::acquisition::IBackend.

8.2.1.2 start()

```
bool qlaib::acquisition::BinReplayBackend::start (
            const BackendConfig & config )  [override], [virtual]
```

Initialise the backend and prepare to emit batches.

Parameters

config	Immutable acquisition parameters.

Returns

true on success; false if initialisation failed.

Implements qlaib::acquisition::IBackend.

8.2.1.3 stop()

```
void qlaib::acquisition::BinReplayBackend::stop ( )  [override], [virtual]
```

Stop acquisition and release any hardware or file handles.

Implements qlaib::acquisition::IBackend.

8.2.2 Member Data Documentation

8.2.2.1 coincWindowPs_

```
long long qlaib::acquisition::BinReplayBackend::coincWindowPs_ {200000}  [private]
```

8.2.2.2 currentSecond_

```
long long qlaib::acquisition::BinReplayBackend::currentSecond_ {0}  [private]
```

8.2.2.3 lastSecond_

```
long long qlaib::acquisition::BinReplayBackend::lastSecond_ {-1}  [private]
```

8.2.2.4 singles_

```
std::map<int, Singles> qlaib::acquisition::BinReplayBackend::singles_  [private]
```

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

- include/qlaib/acquisition/BinReplayBackend.h
- src/acquisition/BinReplayBackend.cpp

8.3 qlaib::data::Coincidence Struct Reference

```
#include <SampleBatch.h>
```

Public Attributes

- std::uint64_t counts
- std::string label

8.3.1 Member Data Documentation

8.3.1.1 counts

```
std::uint64_t qlaib::data::Coincidence::counts
```

8.3.1.2 label

```
std::string qlaib::data::Coincidence::label
```

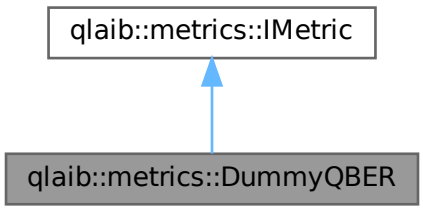
The documentation for this struct was generated from the following file:

- include/qlaib/data/SampleBatch.h

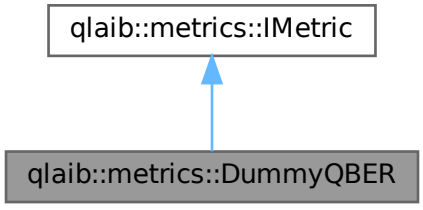
8.4 qlaib::metrics::DummyQBER Class Reference

```
#include <Registry.h>
```

Inheritance diagram for qlaib::metrics::DummyQBER:



Collaboration diagram for qlaib::metrics::DummyQBER:



Public Member Functions

- double compute (const data::SampleBatch &batch) override
- std::string name () const override

Public Member Functions inherited from qlaib::metrics::IMetric

- virtual ~IMetric ()=default

8.4.1 Member Function Documentation

8.4.1.1 compute()

```
double qlaib::metrics::DummyQBER::compute (
            const data::SampleBatch & batch )  [override], [virtual]
```

Implements qlaib::metrics::IMetric.

8.4.1.2 name()

```
std::string qlaib::metrics::DummyQBER::name ( ) const  [inline], [override], [virtual]
```

Implements qlaib::metrics::IMetric.

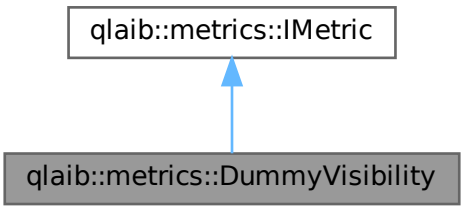
The documentation for this class was generated from the following files:

- include/qlaib/metrics/Registry.h
- src/metrics/Registry.cpp

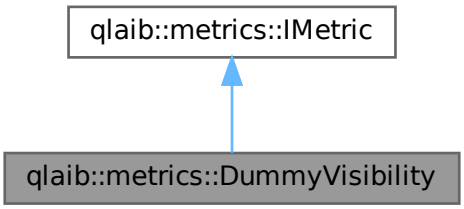
8.5 qlaib::metrics::DummyVisibility Class Reference

```
#include <Registry.h>
```

Inheritance diagram for qlaib::metrics::DummyVisibility:



Collaboration diagram for qlaib::metrics::DummyVisibility:



Public Member Functions

- double compute (const data::SampleBatch &batch) override
- std::string name () const override

Public Member Functions inherited from qlaib::metrics::IMetric

- virtual ~IMetric ()=default

8.5.1 Member Function Documentation

8.5.1.1 compute()

```
double qlaib::metrics::DummyVisibility::compute (
            const data::SampleBatch & batch )  [override], [virtual]
```

Implements qlaib::metrics::IMetric.

8.5.1.2 name()

```
std::string qlaib::metrics::DummyVisibility::name ( ) const  [inline], [override], [virtual]
```

Implements qlaib::metrics::IMetric.

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

- include/qlaib/metrics/Registry.h
- src/metrics/Registry.cpp

8.6 qlaib::core::EventBus< Payload > Class Template Reference

```
#include <EventBus.h>
```

Classes

- struct Subscriber

Public Types

- using Callback = std::function< void(const Payload &)>

Public Member Functions

- void publish (const std::string &topic, const Payload &payload)
- int subscribe (const std::string &topic, Callback cb)

Private Attributes

- std::mutex mutex_
- int nextToken_ {0}
- std::map< std::string, std::vector< Subscriber > > subscribers_

8.6.1 Member Typedef Documentation

8.6.1.1 Callback

```
template<typename Payload >
using qlaib::core::EventBus< Payload >::Callback = std::function<void(const Payload &)>
```

8.6.2 Member Function Documentation

8.6.2.1 publish()

```
template<typename Payload >
void qlaib::core::EventBus< Payload >::publish (
            const std::string & topic,
            const Payload & payload )  [inline]
```

8.6.2.2 subscribe()

```
template<typename Payload >
int qlaib::core::EventBus< Payload >::subscribe (
            const std::string & topic,
            Callback cb )  [inline]
```

8.6.3 Member Data Documentation

8.6.3.1 mutex_

```
template<typename Payload >
std::mutex qlaib::core::EventBus< Payload >::mutex_  [private]
```

8.6.3.2 nextToken_

```
template<typename Payload >
int qlaib::core::EventBus< Payload >::nextToken_ {0}  [private]
```

8.6.3.3 subscribers_

```
template<typename Payload >
std::map<std::string, std::vector<Subscriber> > qlaib::core::EventBus< Payload >::subscribers↩
_  [private]
```

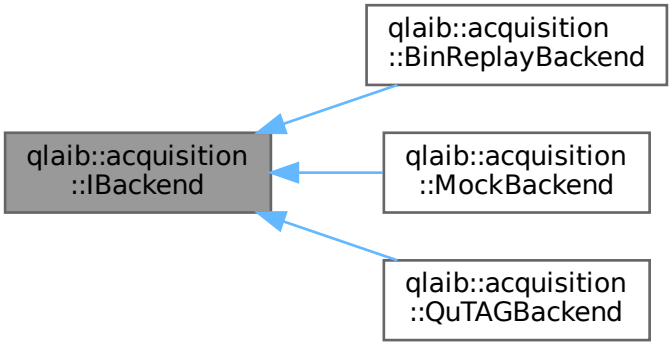
The documentation for this class was generated from the following file:

- include/qlaib/core/EventBus.h

8.7 qlaib::acquisition::IBackend Class Reference

```
#include <IBackend.h>
```

Inheritance diagram for qlaib::acquisition::IBackend:



Public Member Functions

- virtual std::optional< data::SampleBatch > nextBatch ()=0

  Retrieve the next batch of samples, if available.
- virtual bool start (const BackendConfig &config)=0

  Initialise the backend and prepare to emit batches.
- virtual void stop ()=0

  Stop acquisition and release any hardware or file handles.
- virtual ~IBackend ()=default

8.7.1 Constructor & Destructor Documentation

8.7.1.1 ~IBackend()

```
virtual qlaib::acquisition::IBackend::~IBackend ( )  [virtual], [default]
```

8.7.2 Member Function Documentation

8.7.2.1 nextBatch()

```
virtual std::optional< data::SampleBatch > qlaib::acquisition::IBackend::nextBatch ( )  [pure
virtual]
```

Retrieve the next batch of samples, if available.

Returns

A SampleBatch when data is ready; std::nullopt otherwise.

Implemented in qlaib::acquisition::BinReplayBackend, qlaib::acquisition::MockBackend, and qlaib::acquisition::QuTAGBackend.

8.7.2.2 start()

```
virtual bool qlaib::acquisition::IBackend::start (
            const BackendConfig & config )  [pure virtual]
```

Initialise the backend and prepare to emit batches.

Parameters

config	Immutable acquisition parameters.

Returns

true on success; false if initialisation failed.

Implemented in qlaib::acquisition::BinReplayBackend, qlaib::acquisition::MockBackend, and qlaib::acquisition::QuTAGBackend.

8.7.2.3 stop()

```
virtual void qlaib::acquisition::IBackend::stop ( )  [pure virtual]
```

Stop acquisition and release any hardware or file handles.

Implemented in qlaib::acquisition::BinReplayBackend, qlaib::acquisition::MockBackend, and qlaib::acquisition::QuTAGBackend.

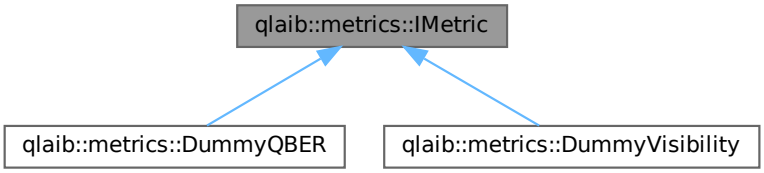
The documentation for this class was generated from the following file:

- include/qlaib/acquisition/IBackend.h

8.8 qlaib::metrics::IMetric Class Reference

`#include <IMetric.h>`

Inheritance diagram for qlaib::metrics::IMetric:



Public Member Functions

- virtual double compute (const data::SampleBatch &batch)=0
- virtual std::string name () const =0
- virtual ~IMetric ()=default

8.8.1 Constructor & Destructor Documentation

8.8.1.1 ~IMetric()

```
virtual qlaib::metrics::IMetric::~IMetric ( )  [virtual], [default]
```

8.8.2 Member Function Documentation

8.8.2.1 compute()

```
virtual double qlaib::metrics::IMetric::compute (
            const data::SampleBatch & batch )  [pure virtual]
```

Implemented in qlaib::metrics::DummyVisibility, and qlaib::metrics::DummyQBER.

8.8.2.2 name()

```
virtual std::string qlaib::metrics::IMetric::name ( ) const  [pure virtual]
```

Implemented in qlaib::metrics::DummyVisibility, and qlaib::metrics::DummyQBER.

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

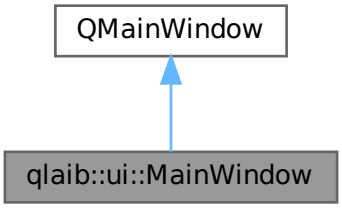
- include/qlaib/metrics/IMetric.h

8.9 qlaib::ui::MainWindow Class Reference

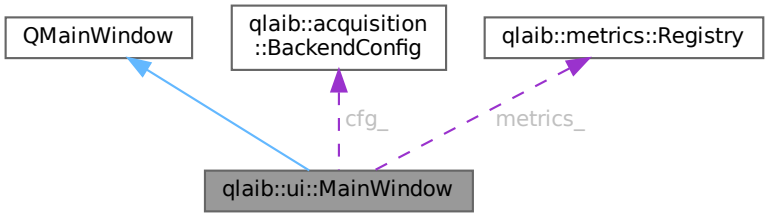
Main GUI window: drives backends and renders live metrics/plots.

`#include <MainWindow.h>`

Inheritance diagram for qlaib::ui::MainWindow:



Collaboration diagram for qlaib::ui::MainWindow:



Public Member Functions

- MainWindow (QWidget ∗parent=nullptr)
- void setMode (const QString &mode)
- void setReplayFile (const QString &file)
- void start ()
- void stop ()

Private Slots

- void addPair ()
- void calibrateAll ()
- void calibrateSelected ()
- void computeHistogram ()
- void exportCsv ()
- void removeSelectedPair ()
- void resetPairs ()
- void tick ()
- void toggleRecording ()

Private Member Functions

- void appendSample (const data::SampleBatch &batch)
- void loadConfig ()
- void refreshPairList (const data::SampleBatch &batch)
- void refreshPairsTable ()
- void saveConfig ()
- void setupUi ()

Private Attributes

- std::unique_ptr< acquisition::IBackend > backend_
- acquisition::BackendConfig cfg_
- double coincidenceWindowPs_ {200.0}
- double histogramWindowPs_ {200.0}
- std::optional< data::SampleBatch > latestBatch_
- metrics::Registry metrics_
- QString mode_ {"live"}
- std::vector< PairSpec > pairs_
- QString replayFile_
- qint64 t0_ms_ {0}
- QTimer timer_

8.9.1 Detailed Description

Main GUI window: drives backends and renders live metrics/plots.

Responsibilities:

- select and start the desired backend (live quTAG, replay, mock)
- pull batches on a timer and update charts/tables
- manage coincidence pairs and calibration shortcuts

8.9.2 Constructor & Destructor Documentation

8.9.2.1 MainWindow()

```
qlaib::ui::MainWindow::MainWindow (
            QWidget * parent = nullptr )  [explicit]
```

8.9.3 Member Function Documentation

8.9.3.1 addPair

```
void qlaib::ui::MainWindow::addPair ( )  [private], [slot]
```

8.9.3.2 appendSample()

```
void qlaib::ui::MainWindow::appendSample (
            const data::SampleBatch & batch )  [private]
```

8.9.3.3 calibrateAll

```
void qlaib::ui::MainWindow::calibrateAll ( )  [private], [slot]
```

8.9.3.4 calibrateSelected

```
void qlaib::ui::MainWindow::calibrateSelected ( )  [private], [slot]
```

8.9.3.5 computeHistogram

```
void qlaib::ui::MainWindow::computeHistogram ( )  [private], [slot]
```

8.9.3.6 exportCsv

```
void qlaib::ui::MainWindow::exportCsv ( )  [private], [slot]
```

8.9.3.7 loadConfig()

```
void qlaib::ui::MainWindow::loadConfig ( )  [private]
```

8.9.3.8 refreshPairList()

```
void qlaib::ui::MainWindow::refreshPairList (
            const data::SampleBatch & batch )  [private]
```

8.9.3.9 refreshPairsTable()

```
void qlaib::ui::MainWindow::refreshPairsTable ( )  [private]
```

8.9.3.10 removeSelectedPair

```
void qlaib::ui::MainWindow::removeSelectedPair ( )  [private], [slot]
```

8.9.3.11 resetPairs

```
void qlaib::ui::MainWindow::resetPairs ( )  [private], [slot]
```

8.9.3.12 saveConfig()

```
void qlaib::ui::MainWindow::saveConfig ( )  [private]
```

8.9.3.13 setMode()

```
void qlaib::ui::MainWindow::setMode (
            const QString & mode )  [inline]
```

8.9.3.14 setReplayFile()

```
void qlaib::ui::MainWindow::setReplayFile (
            const QString & file )  [inline]
```

8.9.3.15 setupUi()

```
void qlaib::ui::MainWindow::setupUi ( )  [private]
```

8.9.3.16 start()

```
void qlaib::ui::MainWindow::start ( )
```

8.9.3.17 stop()

```
void qlaib::ui::MainWindow::stop ( )
```

8.9.3.18 tick

```
void qlaib::ui::MainWindow::tick ( )  [private], [slot]
```

8.9.3.19 toggleRecording

```
void qlaib::ui::MainWindow::toggleRecording ( )  [private], [slot]
```

8.9.4 Member Data Documentation

8.9.4.1 backend_

```
std::unique_ptr<acquisition::IBackend> qlaib::ui::MainWindow::backend_  [private]
```

8.9.4.2 cfg_

```
acquisition::BackendConfig qlaib::ui::MainWindow::cfg_  [private]
```

8.9.4.3 coincidenceWindowPs_

```
double qlaib::ui::MainWindow::coincidenceWindowPs_ {200.0}  [private]
```

8.9.4.4 histogramWindowPs_

```
double qlaib::ui::MainWindow::histogramWindowPs_ {200.0}  [private]
```

8.9.4.5 latestBatch_

```
std::optional<data::SampleBatch> qlaib::ui::MainWindow::latestBatch_  [private]
```

8.9.4.6 metrics_

```
metrics::Registry qlaib::ui::MainWindow::metrics_  [private]
```

8.9.4.7 mode_

```
QString qlaib::ui::MainWindow::mode_ {"live"}  [private]
```

8.9.4.8 pairs_

```
std::vector<PairSpec> qlaib::ui::MainWindow::pairs_  [private]
```

8.9.4.9 replayFile_

```
QString qlaib::ui::MainWindow::replayFile_  [private]
```

8.9.4.10 t0_ms_

```
qint64 qlaib::ui::MainWindow::t0_ms_ {0}  [private]
```

8.9.4.11 timer_

```
QTimer qlaib::ui::MainWindow::timer_  [private]
```

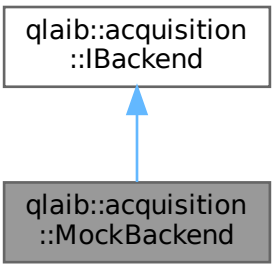
The documentation for this class was generated from the following files:

- include/qlaib/ui/MainWindow.h
- src/ui/MainWindow.cpp

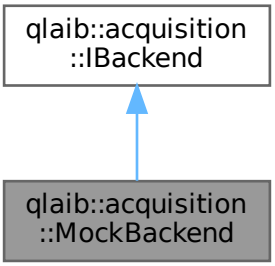
8.10 qlaib::acquisition::MockBackend Class Reference

`#include <MockBackend.h>`

Inheritance diagram for qlaib::acquisition::MockBackend:



Collaboration diagram for qlaib::acquisition::MockBackend:



Public Member Functions

- std::optional< data::SampleBatch > nextBatch () override

  Retrieve the next batch of samples, if available.
- bool start (const BackendConfig &config) override

  Initialise the backend and prepare to emit batches.
- void stop () override

  Stop acquisition and release any hardware or file handles.

Public Member Functions inherited from qlaib::acquisition::IBackend

- virtual ~IBackend ()=default

Private Attributes

- double exposure_ {1.0}
- std::mt19937 rng_ {std::random_device{}()}
- bool running_ {false}
- std::chrono::steady_clock::time_point startTime_

8.10.1 Member Function Documentation

8.10.1.1 nextBatch()

```
std::optional< data::SampleBatch > qlaib::acquisition::MockBackend::nextBatch ( )  [override], [virtual]
```

Retrieve the next batch of samples, if available.

Returns

A SampleBatch when data is ready; std::nullopt otherwise.

Implements qlaib::acquisition::IBackend.

8.10.1.2 start()

```
bool qlaib::acquisition::MockBackend::start (
            const BackendConfig & config )  [override], [virtual]
```

Initialise the backend and prepare to emit batches.

Parameters

config	Immutable acquisition parameters.

Returns

true on success; false if initialisation failed.

Implements qlaib::acquisition::IBackend.

8.10.1.3 stop()

```
void qlaib::acquisition::MockBackend::stop ( )  [override], [virtual]
```

Stop acquisition and release any hardware or file handles.

Implements qlaib::acquisition::IBackend.

8.10.2 Member Data Documentation

8.10.2.1 exposure_

```
double qlaib::acquisition::MockBackend::exposure_ {1.0}  [private]
```

8.10.2.2 rng_

```
std::mt19937 qlaib::acquisition::MockBackend::rng_ {std::random_device{}()}  [private]
```

8.10.2.3 running_

```
bool qlaib::acquisition::MockBackend::running_ {false}  [private]
```

8.10.2.4 startTime_

```
std::chrono::steady_clock::time_point qlaib::acquisition::MockBackend::startTime_  [private]
```

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

- include/qlaib/acquisition/MockBackend.h
- src/acquisition/MockBackend.cpp

8.11 qlaib::ui::PairSpec Struct Reference

User-configurable coincidence pair (channel A/B and relative delay).

```
#include <MainWindow.h>
```

Public Attributes

- int chA
  1-based channel index A.
- int chB
  1-based channel index B.
- long long delayPs
  Optional calibrated delay (picoseconds).
- QString label
  Display name shown in tables/plots.

8.11.1 Detailed Description

User-configurable coincidence pair (channel A/B and relative delay).

8.11.2 Member Data Documentation

8.11.2.1 chA

```
int qlaib::ui::PairSpec::chA
```

1-based channel index A.

8.11.2.2 chB

```
int qlaib::ui::PairSpec::chB
```

1-based channel index B.

8.11.2.3 delayPs

```
long long qlaib::ui::PairSpec::delayPs
```

Optional calibrated delay (picoseconds).

8.11.2.4 label

```
QString qlaib::ui::PairSpec::label
```

Display name shown in tables/plots.

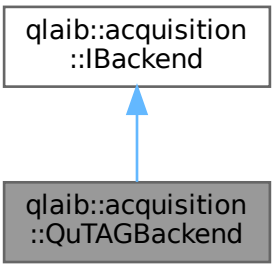
The documentation for this struct was generated from the following file:

- include/qlaib/ui/MainWindow.h

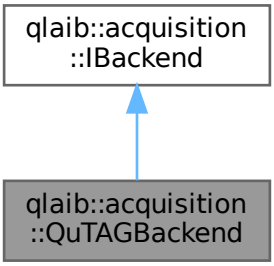
8.12 qlaib::acquisition::QuTAGBackend Class Reference

```
#include <QuTAGBackend.h>
```

Inheritance diagram for qlaib::acquisition::QuTAGBackend:



Collaboration diagram for qlaib::acquisition::QuTAGBackend:



Public Member Functions

- std::optional< data::SampleBatch > nextBatch () override

  Retrieve the next batch of samples, if available.
- bool start (const BackendConfig &config) override

  Initialise the backend and prepare to emit batches.
- bool startRecording (const std::string &filepath, bool compressed=false)
- void stop () override

  Stop acquisition and release any hardware or file handles.
- void stopRecording ()

Public Member Functions inherited from qlaib::acquisition::IBackend

- virtual ~IBackend ()=default

Private Attributes

- int bufferSize_ {200000}
- std::vector< Uint8 > chBuffer_
- long long coincWindowPs_ {200000}
- bool connected_ {false}
- double exposureMs_ {1000.0}
- bool recording_ {false}
- std::vector< Int64 > tsBuffer_

8.12.1 Member Function Documentation

8.12.1.1 nextBatch()

```
std::optional< data::SampleBatch > qlaib::acquisition::QuTAGBackend::nextBatch ( )  [override],
[virtual]
```

Retrieve the next batch of samples, if available.

Returns

A SampleBatch when data is ready; std::nullopt otherwise.

Implements qlaib::acquisition::IBackend.

8.12.1.2 start()

```
bool qlaib::acquisition::QuTAGBackend::start (
            const BackendConfig & config )  [override], [virtual]
```

Initialise the backend and prepare to emit batches.

Parameters

config	Immutable acquisition parameters.

Returns

true on success; false if initialisation failed.

Implements qlaib::acquisition::IBackend.

8.12.1.3 startRecording()

```
bool qlaib::acquisition::QuTAGBackend::startRecording (
            const std::string & filepath,
            bool compressed = false )
```

8.12.1.4 stop()

```
void qlaib::acquisition::QuTAGBackend::stop ( )  [override], [virtual]
```

Stop acquisition and release any hardware or file handles.

Implements qlaib::acquisition::IBackend.

8.12.1.5 stopRecording()

```
void qlaib::acquisition::QuTAGBackend::stopRecording ( )
```

8.12.2 Member Data Documentation

8.12.2.1 bufferSize_

```
int qlaib::acquisition::QuTAGBackend::bufferSize_ {200000}  [private]
```

8.12.2.2 chBuffer_

```
std::vector<Uint8> qlaib::acquisition::QuTAGBackend::chBuffer_  [private]
```

8.12.2.3 coincWindowPs_

```
long long qlaib::acquisition::QuTAGBackend::coincWindowPs_ {200000}  [private]
```

8.12.2.4 connected_

```
bool qlaib::acquisition::QuTAGBackend::connected_ {false}  [private]
```

8.12.2.5 exposureMs_

```
double qlaib::acquisition::QuTAGBackend::exposureMs_ {1000.0}  [private]
```

8.12.2.6 recording_

```
bool qlaib::acquisition::QuTAGBackend::recording_ {false}  [private]
```

8.12.2.7 tsBuffer_

```
std::vector<Int64> qlaib::acquisition::QuTAGBackend::tsBuffer_  [private]
```

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

- include/qlaib/acquisition/QuTAGBackend.h
- src/acquisition/QuTAGBackend.cpp

8.13 qlaib::metrics::Registry Class Reference

```
#include <Registry.h>
```

Public Member Functions

- void computeAll (data::SampleBatch &batch) const
- void registerMetric (std::unique_ptr< IMetric > metric)

Private Attributes

- std::vector< std::unique_ptr< IMetric > > metrics_

8.13.1 Member Function Documentation

8.13.1.1 computeAll()

```
void qlaib::metrics::Registry::computeAll (
            data::SampleBatch & batch ) const
```

8.13.1.2 registerMetric()

```
void qlaib::metrics::Registry::registerMetric (
            std::unique_ptr< IMetric > metric )
```

8.13.2 Member Data Documentation

8.13.2.1 metrics_

```
std::vector<std::unique_ptr<IMetric> > qlaib::metrics::Registry::metrics_  [private]
```

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

- include/qlaib/metrics/Registry.h
- src/metrics/Registry.cpp

8.14 qlaib::data::SampleBatch Struct Reference

```
#include <SampleBatch.h>
```

Public Attributes

- std::vector< Coincidence > coincidences
- std::unordered_map< std::string, double > metrics
- std::vector< std::uint64_t > singles
- std::chrono::steady_clock::time_point timestamp
- std::vector< std::vector< long long > > timestamps_ps

8.14.1 Member Data Documentation

8.14.1.1 coincidences

```
std::vector<Coincidence> qlaib::data::SampleBatch::coincidences
```

8.14.1.2 metrics

```
std::unordered_map<std::string, double> qlaib::data::SampleBatch::metrics
```

8.14.1.3 singles

```
std::vector<std::uint64_t> qlaib::data::SampleBatch::singles
```

8.14.1.4 timestamp

```
std::chrono::steady_clock::time_point qlaib::data::SampleBatch::timestamp
```

8.14.1.5 timestamps_ps

```
std::vector<std::vector<long long> > qlaib::data::SampleBatch::timestamps_ps
```

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

- include/qlaib/data/SampleBatch.h

8.15 qlaib::core::EventBus< Payload >::Subscriber Struct Reference

Public Attributes

- Callback cb
- int token

8.15.1 Member Data Documentation

8.15.1.1 cb

```
template<typename Payload >
Callback qlaib::core::EventBus< Payload >::Subscriber::cb
```

8.15.1.2 token

```
template<typename Payload >
int qlaib::core::EventBus< Payload >::Subscriber::token
```

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

- include/qlaib/core/EventBus.h

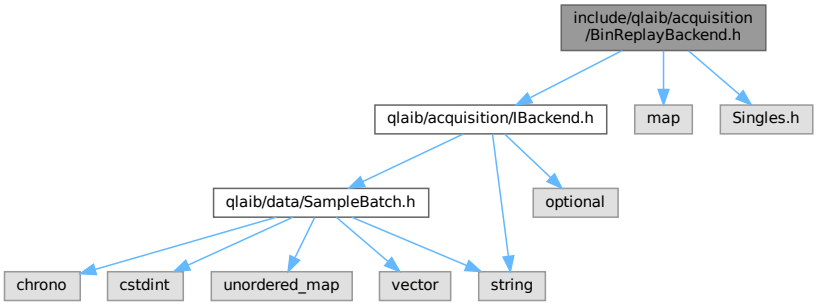
Chapter 9

File Documentation

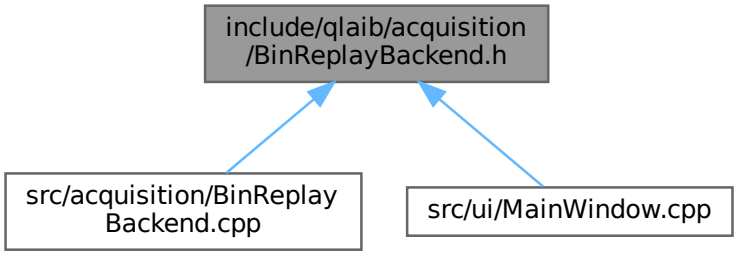
9.1 include/qlaib/acquisition/BinReplayBackend.h File Reference

```
#include "qlaib/acquisition/IBackend.h"
#include <map>
#include "Singles.h"
```

Include dependency graph for BinReplayBackend.h:



This graph shows which files directly or indirectly include this file:



Classes

- class qlaib::acquisition::BinReplayBackend

Namespaces

- namespace qlaib
- namespace qlaib::acquisition

9.2 BinReplayBackend.h

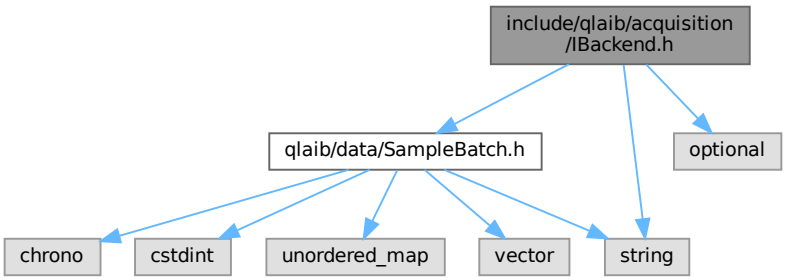
Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include "qlaib/acquisition/IBackend.h"
00004 #include <map>
00005 #include "Singles.h"
00006
00007 namespace qlaib::acquisition {
00008
00009 // Replays a saved BIN file (quTAG timestamps) using coincfinder utilities.
00010 class BinReplayBackend : public IBackend {
00011 public:
00012   bool start(const BackendConfig &config) override;
00013   void stop() override;
00014   std::optional<data::SampleBatch> nextBatch() override;
00015
00016 private:
00017   std::map<int, Singles> singles_;
00018   long long currentSecond_{0};
00019   long long lastSecond_{-1};
00020   long long coincWindowPs_{200000};
00021 };
00022
00023 } // namespace qlaib::acquisition
```

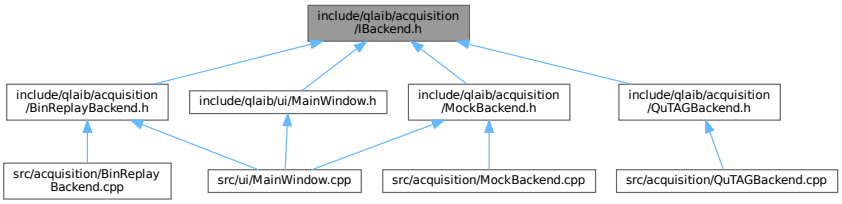
9.3 include/qlaib/acquisition/IBackend.h File Reference

```
#include "qlaib/data/SampleBatch.h"
#include <optional>
#include <string>
```

Include dependency graph for IBackend.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct qlaib::acquisition::BackendConfig

  Immutable configuration shared by all acquisition backends.
- class qlaib::acquisition::IBackend

Namespaces

- namespace qlaib
- namespace qlaib::acquisition

9.4 IBackend.h

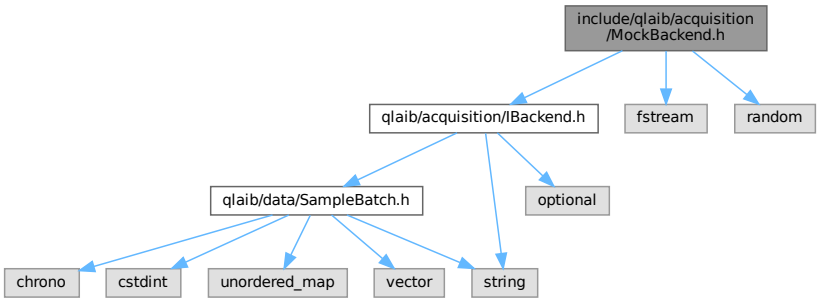
Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include "qlaib/data/SampleBatch.h"
00004 #include <optional>
00005 #include <string>
00006
00007 namespace qlaib::acquisition {
00008
00016 struct BackendConfig {
00018   double exposureSeconds{1.0};
00020   bool useMock{true};
00022   std::string replayFile;
00024   long long coincidenceWindowPs{200000}; // 200 ps default
00026   int timestampBufferSize{200000};
00027 };
00028
00029 class IBackend {
00030 public:
00031   virtual ~IBackend() = default;
00032
00038   virtual bool start(const BackendConfig &config) = 0;
00039
00041   virtual void stop() = 0;
00042
00047   virtual std::optional<data::SampleBatch> nextBatch() = 0;
00048 };
00049
00050 } // namespace qlaib::acquisition
```

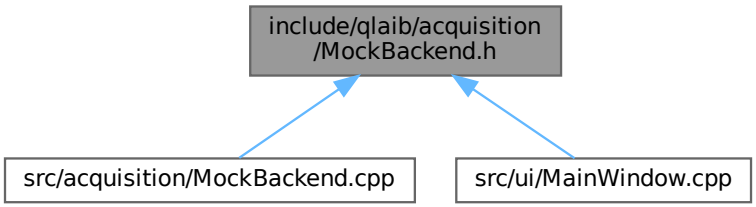
9.5 include/qlaib/acquisition/MockBackend.h File Reference

```
#include "qlaib/acquisition/IBackend.h"
#include <fstream>
#include <random>
```

Include dependency graph for MockBackend.h:



This graph shows which files directly or indirectly include this file:



Classes

- class qlaib::acquisition::MockBackend

Namespaces

- namespace qlaib
- namespace qlaib::acquisition

9.6 MockBackend.h

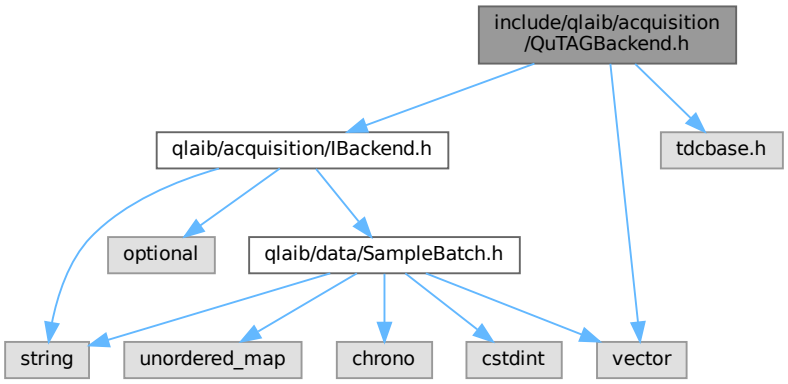
Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include "qlaib/acquisition/IBackend.h"
00004 #include <fstream>
00005 #include <random>
00006
00007 namespace qlaib::acquisition {
00008
00009 // Mock backend: generates synthetic singles/coincidences for UI/dev work.
00010 class MockBackend : public IBackend {
00011 public:
00012   bool start(const BackendConfig &config) override;
00013   void stop() override;
00014   std::optional<data::SampleBatch> nextBatch() override;
00015
00016 private:
00017   std::mt19937 rng_{std::random_device{}()};
00018   std::chrono::steady_clock::time_point startTime_;
00019   double exposure_{1.0};
00020   bool running_{false};
00021 };
00022
00023 } // namespace qlaib::acquisition
```

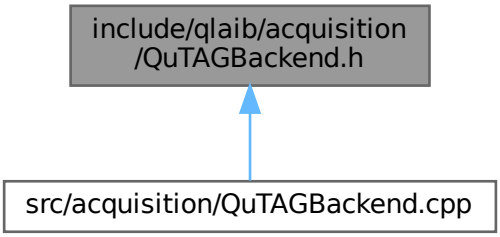
9.7 include/qlaib/acquisition/QuTAGBackend.h File Reference

```
#include "qlaib/acquisition/IBackend.h"
#include "tdcbase.h"
#include <vector>
```

Include dependency graph for QuTAGBackend.h:



This graph shows which files directly or indirectly include this file:



Classes

- class qlaib::acquisition::QuTAGBackend

Namespaces

- namespace qlaib
- namespace qlaib::acquisition

9.8 QuTAGBackend.h

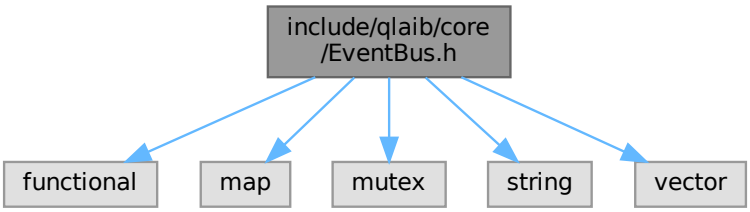
Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include "qlaib/acquisition/IBackend.h"
00004 #include "tdcbase.h"
00005 #include <vector>
00006
00007 namespace qlaib::acquisition {
00008
00009 // Minimal backend that initializes quTAG via libtdcbase and can write .bin captures.
00010 class QuTAGBackend : public IBackend {
00011 public:
00012   bool start(const BackendConfig &config) override;
00013   void stop() override;
00014   std::optional<data::SampleBatch> nextBatch() override;
00015
00016   // Start writing timestamps to a binary file (compressed=false => FORMAT_BINARY).
00017   bool startRecording(const std::string &filepath, bool compressed = false);
00018   void stopRecording();
00019
00020 private:
00021   std::vector<Int64> tsBuffer_;
00022   std::vector<Uint8> chBuffer_;
00023   bool connected_{false};
00024   bool recording_{false};
00025   double exposureMs_{1000.0};
00026   long long coincWindowPs_{200000};
00027   int bufferSize_{200000};
00028 };
00029
00030 } // namespace qlaib::acquisition
```

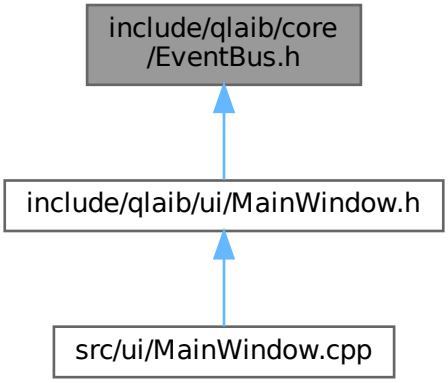
9.9 include/qlaib/core/EventBus.h File Reference

```
#include <functional>
#include <map>
#include <mutex>
#include <string>
#include <vector>
```

Include dependency graph for EventBus.h:



This graph shows which files directly or indirectly include this file:



Classes

- class qlaib::core::EventBus< Payload >
- struct qlaib::core::EventBus< Payload >::Subscriber

Namespaces

- namespace qlaib
- namespace qlaib::core

9.10 EventBus.h

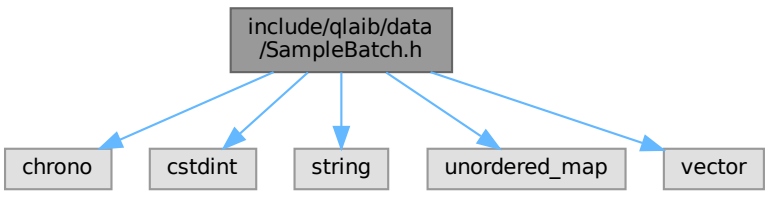
Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include <functional>
00004 #include <map>
00005 #include <mutex>
00006 #include <string>
00007 #include <vector>
00008
00009 namespace qlaib::core {
00010
00011 // Lightweight pub-sub bus used to decouple acquisition threads from the UI.
00012 template <typename Payload>
00013 class EventBus {
00014 public:
00015   using Callback = std::function<void(const Payload &)>;
00016
00017   int subscribe(const std::string &topic, Callback cb) {
00018     std::scoped_lock lk(mutex_);
00019     int token = nextToken_++;
00020     subscribers_[topic].push_back({token, std::move(cb)});
00021     return token;
00022   }
00023
00024   void publish(const std::string &topic, const Payload &payload) {
00025     std::vector<Callback> cbs;
00026     {
00027       std::scoped_lock lk(mutex_);
00028       auto it = subscribers_.find(topic);
00029       if (it == subscribers_.end())
00030         return;
00031       for (auto &sub : it->second)
00032         cbs.push_back(sub.cb);
00033     }
00034     for (auto &cb : cbs)
00035       cb(payload);
00036   }
00037
00038 private:
00039   struct Subscriber {
00040     int token;
00041     Callback cb;
00042   };
00043   std::map<std::string, std::vector<Subscriber>> subscribers_;
00044   int nextToken_{0};
00045   std::mutex mutex_;
00046 };
00047
00048 } // namespace qlaib::core
```

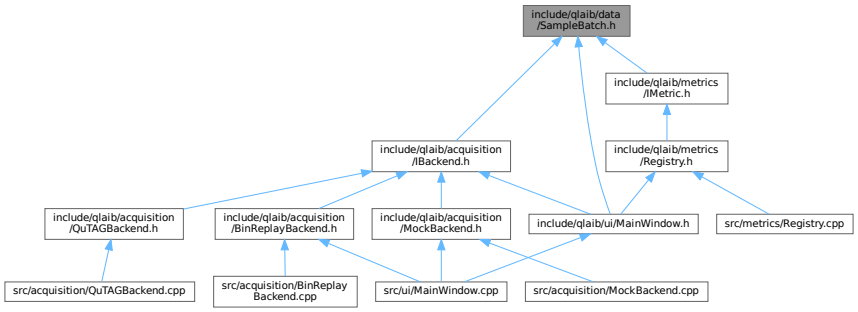
9.11 include/qlaib/data/SampleBatch.h File Reference

```
#include <chrono>
#include <cstdint>
#include <string>
#include <unordered_map>
#include <vector>
```

Include dependency graph for SampleBatch.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct qlaib::data::Coincidence
- struct qlaib::data::SampleBatch

Namespaces

- namespace qlaib
- namespace qlaib::data

9.12 SampleBatch.h

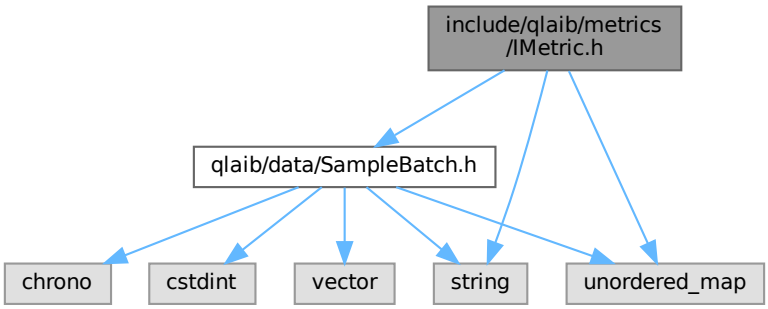
Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include <chrono>
00004 #include <cstdint>
00005 #include <string>
00006 #include <unordered_map>
00007 #include <vector>
00008
00009 namespace qlaib::data {
00010
00011 struct Coincidence {
00012   std::string label;
00013   std::uint64_t counts;
00014 };
00015
00016 struct SampleBatch {
00017   std::chrono::steady_clock::time_point timestamp;
00018   std::vector<std::uint64_t> singles;              // per-channel counts
00019   std::vector<Coincidence> coincidences;           // labeled coincidences
00020   std::unordered_map<std::string, double> metrics;  // QBER, visibility, etc.
00021   std::vector<std::vector<long long» timestamps_ps; // raw timestamps per channel
00022 };
00023
00024 } // namespace qlaib::data
```

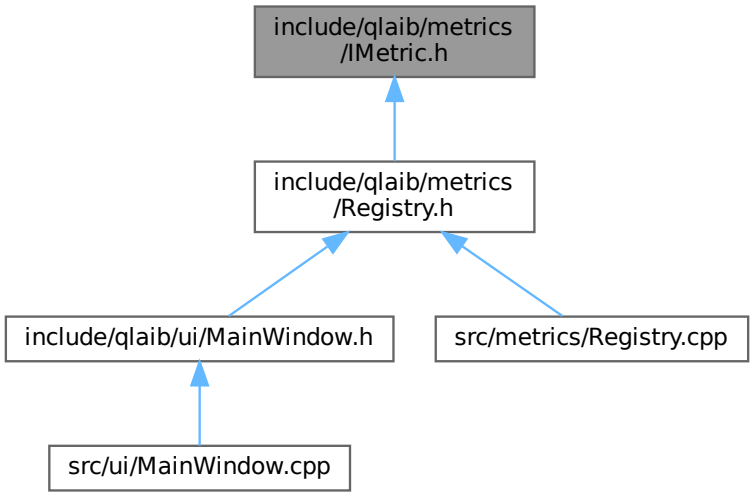
9.13 include/qlaib/metrics/IMetric.h File Reference

```
#include "qlaib/data/SampleBatch.h"
#include <string>
#include <unordered_map>
```

Include dependency graph for IMetric.h:



This graph shows which files directly or indirectly include this file:



Classes

- class qlaib::metrics::IMetric

Namespaces

- namespace qlaib
- namespace qlaib::metrics

9.14 IMetric.h

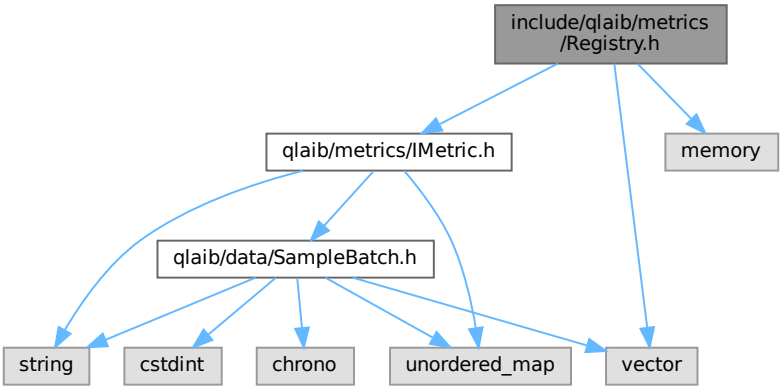
Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include "qlaib/data/SampleBatch.h"
00004 #include <string>
00005 #include <unordered_map>
00006
00007 namespace qlaib::metrics {
00008
00009 class IMetric {
00010 public:
00011   virtual ~IMetric() = default;
00012   virtual std::string name() const = 0;
00013   virtual double compute(const data::SampleBatch &batch) = 0;
00014 };
00015
00016 } // namespace qlaib::metrics
```

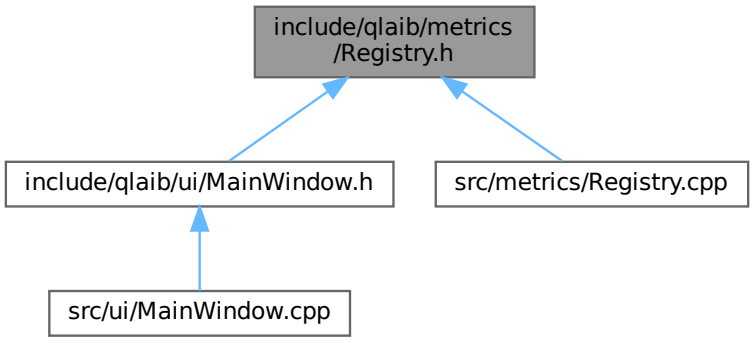
9.15 include/qlaib/metrics/Registry.h File Reference

```
#include "qlaib/metrics/IMetric.h"
#include <memory>
#include <vector>
```

Include dependency graph for Registry.h:



This graph shows which files directly or indirectly include this file:



Classes

- class qlaib::metrics::DummyQBER
- class qlaib::metrics::DummyVisibility
- class qlaib::metrics::Registry

Namespaces

- namespace qlaib
- namespace qlaib::metrics

9.16 Registry.h

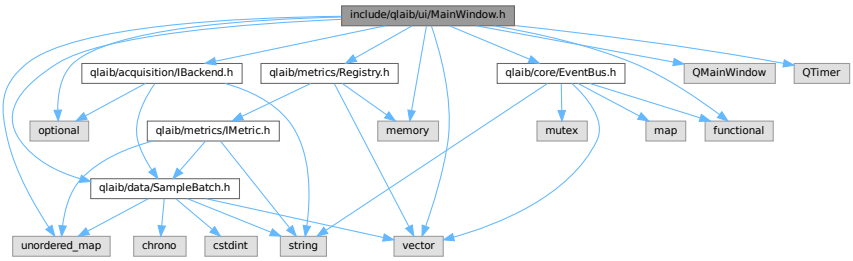
Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include "qlaib/metrics/IMetric.h"
00004 #include <memory>
00005 #include <vector>
00006
00007 namespace qlaib::metrics {
00008
00009 class Registry {
00010 public:
00011   void registerMetric(std::unique_ptr<IMetric> metric);
00012   void computeAll(data::SampleBatch &batch) const;
00013
00014 private:
00015   std::vector<std::unique_ptr<IMetric» metrics_;
00016 };
00017
00018 class DummyVisibility final : public IMetric {
00019 public:
00020   std::string name() const override { return "visibility"; }
00021   double compute(const data::SampleBatch &batch) override;
00022 };
00023
00024 class DummyQBER final : public IMetric {
00025 public:
00026   std::string name() const override { return "qber"; }
00027   double compute(const data::SampleBatch &batch) override;
00028 };
00029
00030 } // namespace qlaib::metrics
```

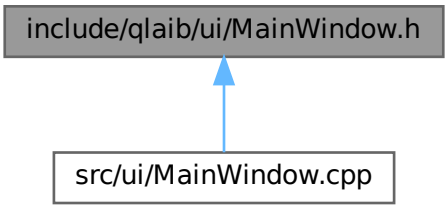
9.17 include/qlaib/ui/MainWindow.h File Reference

```
#include "qlaib/acquisition/IBackend.h"
#include "qlaib/core/EventBus.h"
#include "qlaib/data/SampleBatch.h"
#include "qlaib/metrics/Registry.h"
#include <QMainWindow>
#include <QTimer>
#include <memory>
#include <optional>
#include <functional>
#include <unordered_map>
#include <vector>
```

Include dependency graph for MainWindow.h:



This graph shows which files directly or indirectly include this file:



Classes

- class qlaib::ui::MainWindow

  Main GUI window: drives backends and renders live metrics/plots.
- struct qlaib::ui::PairSpec

  User-configurable coincidence pair (channel A/B and relative delay).

Namespaces

- namespace qlaib
- namespace qlaib::ui

9.18 MainWindow.h

Go to the documentation of this file.

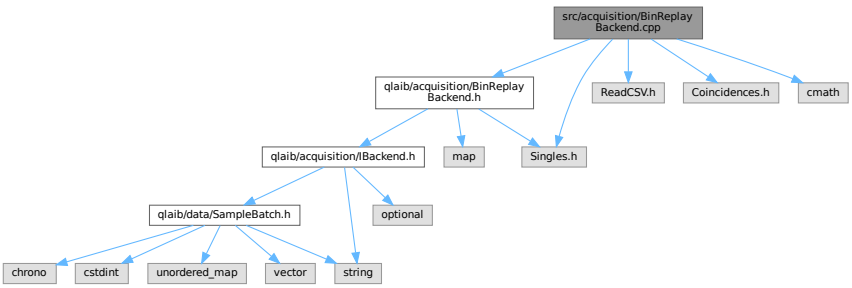
```
00001 #pragma once
00002
00003 #include "qlaib/acquisition/IBackend.h"
00004 #include "qlaib/core/EventBus.h"
00005 #include "qlaib/data/SampleBatch.h"
00006 #include "qlaib/metrics/Registry.h"
00007 #include <QMainWindow>
00008 #include <QTimer>
00009 #include <memory>
00010 #include <optional>
00011 #include <functional>
00012 #include <unordered_map>
00013 #include <vector>
00014
00015 #ifdef QQL_ENABLE_CHARTS
00016 QT_BEGIN_NAMESPACE
00017 class QChartView;
00018 class QLineSeries;
00019 class QValueAxis;
00020 class QChart;
00021 class QTabWidget;
00022 class QComboBox;
00023 class QDoubleSpinBox;
00024 class QPushButton;
00025 class QShortcut;
00026 class QTableWidget;
00027 class QVBoxLayout;
00028 QT_END_NAMESPACE
00029 #endif
00030
00031 namespace qlaib::ui {
00032
00034 struct PairSpec {
00035   QString label;
00036   int chA;
00037   int chB;
00038   long long delayPs;
00039 };
00040
00049 class MainWindow : public QMainWindow {
00050   Q_OBJECT
00051 public:
00052   explicit MainWindow(QWidget *parent = nullptr);
00053   void start();
00054   void stop();
00055
00056   // CLI hooks
00057   void setMode(const QString &mode) { mode_ = mode; }
00058   void setReplayFile(const QString &file) { replayFile_ = file; }
00059
00060 private slots:
00061   void tick();
00062   void computeHistogram();
00063   void exportCsv();
00064   void toggleRecording();
00065   void addPair();
00066   void removeSelectedPair();
00067   void resetPairs();
00068   void calibrateSelected();
00069   void calibrateAll();
00070
00071 private:
00072   void setupUi();
00073 #ifdef QQL_ENABLE_CHARTS
00074   void setupChartTabs();
00075   void setupHistogramTab();
00076   void setupPairsUI();
00077   void connectShortcut(Qt::Key key, std::function<void()> slot);
00078 #endif
00079   void appendSample(const data::SampleBatch &batch);
00080   void refreshPairList(const data::SampleBatch &batch);
00081   void refreshPairsTable();
00082   void saveConfig();
00083   void loadConfig();
00084
00085   acquisition::BackendConfig cfg_;
00086   std::unique_ptr<acquisition::IBackend> backend_;
00087   metrics::Registry metrics_;
00088   QTimer timer_;
00089   std::optional<data::SampleBatch> latestBatch_;
00090   std::vector<PairSpec> pairs_;
00091   QString mode_{"live"};
```

```
00092   QString replayFile_;
00093   double histogramWindowPs_{200.0};
00094   double coincidenceWindowPs_{200.0};
00095
00096 #ifdef QQL_ENABLE_CHARTS
00097   QVBoxLayout *mainLayout_{nullptr};
00098   QTabWidget *tabs_{nullptr};
00099   QChartView *singlesView_{nullptr};
00100   QChartView *coincView_{nullptr};
00101   QChartView *metricsView_{nullptr};
00102   QChartView *histView_{nullptr};
00103   QWidget *histTab_{nullptr};
00104   QChart *singlesChart_{nullptr};
00105   QChart *coincChart_{nullptr};
00106   QChart *metricsChart_{nullptr};
00107   QChart *histChart_{nullptr};
00108   QValueAxis *singlesAxisX_{nullptr};
00109   QValueAxis *singlesAxisY_{nullptr};
00110   QValueAxis *coincAxisX_{nullptr};
00111   QValueAxis *coincAxisY_{nullptr};
00112   QValueAxis *metricsAxisX_{nullptr};
00113   QValueAxis *metricsAxisY_{nullptr};
00114   QValueAxis *histAxisX_{nullptr};
00115   QValueAxis *histAxisY_{nullptr};
00116   QComboBox *pairBox_{nullptr};
00117   QDoubleSpinBox *windowSpin_{nullptr};
00118   QDoubleSpinBox *startSpin_{nullptr};
00119   QDoubleSpinBox *endSpin_{nullptr};
00120   QDoubleSpinBox *stepSpin_{nullptr};
00121   QDoubleSpinBox *exposureSpin_{nullptr};
00122   QDoubleSpinBox *coincWindowSpin_{nullptr};
00123   QTableWidget *pairsTable_{nullptr};
00124   QPushButton *histBtn_{nullptr};
00125   QPushButton *recordBtn_{nullptr};
00126   bool fillingPairsTable_{false};
00127   std::unordered_map<QString, QLineSeries *> singlesSeries_;
00128   std::unordered_map<QString, QLineSeries *> coincSeries_;
00129   std::unordered_map<QString, QLineSeries *> metricSeries_;
00130   int historyLen_{400};
00131 #endif
00132   qint64 t0_ms_{0};
00133 };
00134
00135 } // namespace qlaib::ui
```

9.19 src/acquisition/BinReplayBackend.cpp File Reference

```
#include "qlaib/acquisition/BinReplayBackend.h"
#include "ReadCSV.h"
#include "Coincidences.h"
#include "Singles.h"
#include <cmath>
```

Include dependency graph for BinReplayBackend.cpp:



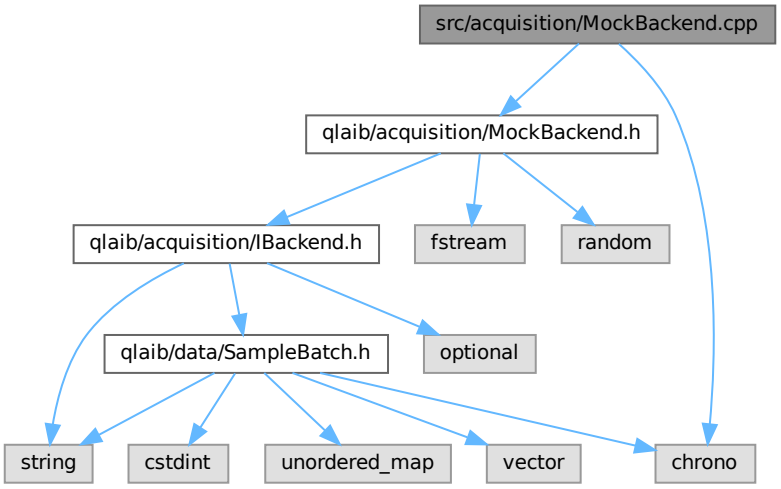
Namespaces

- namespace qlaib
- namespace qlaib::acquisition

9.20 src/acquisition/MockBackend.cpp File Reference

```
#include "qlaib/acquisition/MockBackend.h"
#include <chrono>
```

Include dependency graph for MockBackend.cpp:



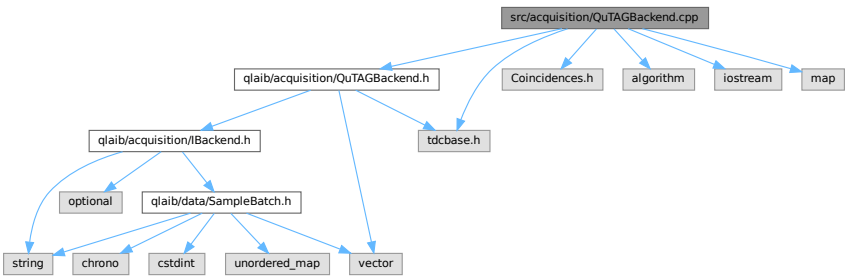
Namespaces

- namespace qlaib
- namespace qlaib::acquisition

9.21 src/acquisition/QuTAGBackend.cpp File Reference

```
#include "qlaib/acquisition/QuTAGBackend.h"
#include "Coincidences.h"
#include "tdcbase.h"
#include <algorithm>
#include <iostream>
#include <map>
```

Include dependency graph for QuTAGBackend.cpp:



Namespaces

- namespace qlaib
- namespace qlaib::acquisition
- namespace qlaib::acquisition::anonymous_namespace{QuTAGBackend.cpp}

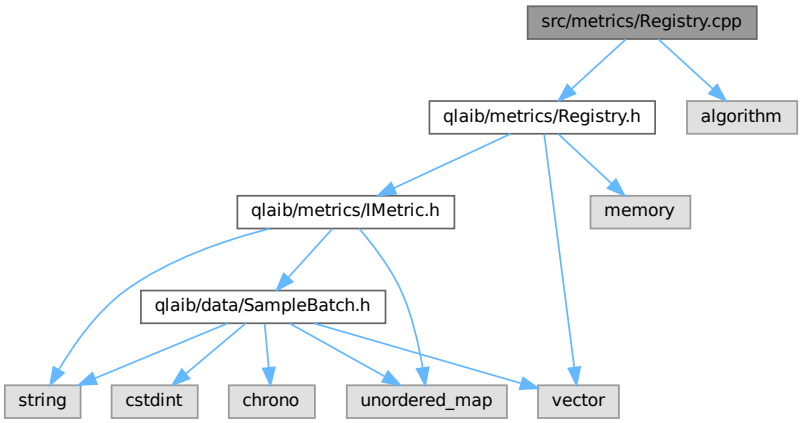
Variables

- constexpr int qlaib::acquisition::anonymous_namespace{QuTAGBackend.cpp}::kDefaultChannelMask = 0x↩ FF

9.22 src/metrics/Registry.cpp File Reference

```
#include "qlaib/metrics/Registry.h"
#include <algorithm>
```

Include dependency graph for Registry.cpp:



Namespaces

- namespace qlaib
- namespace qlaib::metrics

9.23 src/ui/MainWindow.cpp File Reference

```
#include "qlaib/ui/MainWindow.h"
#include "Coincidences.h"
#include "ReadCSV.h"
#include "qlaib/acquisition/BinReplayBackend.h"
#include "qlaib/acquisition/MockBackend.h"
#include <vector>
#include <QFile>
```

```
#include <QFileDialog>
#include <QHBoxLayout>
#include <QHeaderView>
#include <QJsonArray>
#include <QJsonDocument>
#include <QJsonObject>
#include <QLabel>
#include <QDebug>
#include <QPushButton>
#include <QShortcut>
#include <QSpinBox>
#include <QStandardPaths>
#include <QStatusBar>
#include <QTabWidget>
#include <QTableWidget>
#include <QTextStream>
#include <QVBoxLayout>
#include <QtWidgets>
#include <cstdlib>
#include <optional>
#include <span>
#include <unordered_map>
```

Include dependency graph for MainWindow.cpp:

Namespaces

- namespace qlaib
- namespace qlaib::ui

Functions

- long long findBestDelayPicoseconds (std::span< const long long > reference, std::span< const long long > target, long long coincWindowPs, long long delayStartPs, long long delayEndPs, long long delayStepPs, std::vector< std::pair< float, int > > *scratchResults)

9.23.1 Function Documentation

9.23.1.1 findBestDelayPicoseconds()

```
long long findBestDelayPicoseconds (
            std::span< const long long > reference,
            std::span< const long long > target,
            long long coincWindowPs,
            long long delayStartPs,
            long long delayEndPs,
            long long delayStepPs,
            std::vector< std::pair< float, int > > * scratchResults )
```

9.24 /home/bogfootlj/Documents/PhDCode/QLaibLibCpp/docs/↩ ARCHITECTURE.md File Reference

9.25 /home/bogfootlj/Documents/PhDCode/QLaibLibCpp/docs/↩ USAGE.md File Reference

Index

/home/bogfootlj/Documents/PhDCode/QLaibLibCpp/docs/ARCHITECTURE.md, 59
/home/bogfootlj/Documents/PhDCode/QLaibLibCpp/docs/USAGE.md, 59
~IBackend
    qlaib::acquisition::IBackend, 25
~IMetric
    qlaib::metrics::IMetric, 26

addPair
    qlaib::ui::MainWindow, 28
appendSample
    qlaib::ui::MainWindow, 28

backend_
    qlaib::ui::MainWindow, 30
bufferSize_
    qlaib::acquisition::QuTAGBackend, 37

calibrateAll
    qlaib::ui::MainWindow, 29
calibrateSelected
    qlaib::ui::MainWindow, 29
Callback
    qlaib::core::EventBus< Payload >, 23
cb
    qlaib::core::EventBus< Payload >::Subscriber, 40
cfg_
    qlaib::ui::MainWindow, 30
chA
    qlaib::ui::PairSpec, 35
chB
    qlaib::ui::PairSpec, 35
chBuffer_
    qlaib::acquisition::QuTAGBackend, 37
coincidences
    qlaib::data::SampleBatch, 39
coincidenceWindowPs
    qlaib::acquisition::BackendConfig, 15
coincidenceWindowPs_
    qlaib::ui::MainWindow, 31
coincWindowPs_
    qlaib::acquisition::BinReplayBackend, 19
    qlaib::acquisition::QuTAGBackend, 37
compute
    qlaib::metrics::DummyQBER, 21
    qlaib::metrics::DummyVisibility, 22
    qlaib::metrics::IMetric, 26
computeAll
    qlaib::metrics::Registry, 38
computeHistogram
    qlaib::ui::MainWindow, 29
connected_
    qlaib::acquisition::QuTAGBackend, 38
counts
    qlaib::data::Coincidence, 20
currentSecond_
    qlaib::acquisition::BinReplayBackend, 19

delayPs
    qlaib::ui::PairSpec, 35

exportCsv
    qlaib::ui::MainWindow, 29
exposure_
    qlaib::acquisition::MockBackend, 34
exposureMs_
    qlaib::acquisition::QuTAGBackend, 38
exposureSeconds
    qlaib::acquisition::BackendConfig, 15

findBestDelayPicoseconds
    MainWindow.cpp, 58

histogramWindowPs_
    qlaib::ui::MainWindow, 31

include/qlaib/acquisition/BinReplayBackend.h, 41, 42
include/qlaib/acquisition/IBackend.h, 42, 43
include/qlaib/acquisition/MockBackend.h, 44, 45
include/qlaib/acquisition/QuTAGBackend.h, 45, 46
include/qlaib/core/EventBus.h, 47, 48
include/qlaib/data/SampleBatch.h, 48, 49
include/qlaib/metrics/IMetric.h, 50, 51
include/qlaib/metrics/Registry.h, 51, 52
include/qlaib/ui/MainWindow.h, 53, 54

kDefaultChannelMask
    qlaib::acquisition::anonymous_namespace{QuTAGBackend.cpp}, 14

label
    qlaib::data::Coincidence, 20
    qlaib::ui::PairSpec, 35
lastSecond_
    qlaib::acquisition::BinReplayBackend, 19
latestBatch_
    qlaib::ui::MainWindow, 31
loadConfig
    qlaib::ui::MainWindow, 29

MainWindow
qlaib::ui::MainWindow, 28
MainWindow.cpp
findBestDelayPicoseconds, 58
metrics
qlaib::data::SampleBatch, 39
metrics_
qlaib::metrics::Registry, 39
qlaib::ui::MainWindow, 31
mode_
qlaib::ui::MainWindow, 31
mutex_
qlaib::core::EventBus< Payload >, 23

name
qlaib::metrics::DummyQBER, 21
qlaib::metrics::DummyVisibility, 22
qlaib::metrics::IMetric, 26
nextBatch
qlaib::acquisition::BinReplayBackend, 17
qlaib::acquisition::IBackend, 25
qlaib::acquisition::MockBackend, 33
qlaib::acquisition::QuTAGBackend, 36
nextToken_
qlaib::core::EventBus< Payload >, 23

pairs_
qlaib::ui::MainWindow, 31
publish
qlaib::core::EventBus< Payload >, 23

qlaib, 13
qlaib::acquisition, 13
qlaib::acquisition::anonymous_namespace{QuTAGBackend.cpp}, 13
kDefaultChannelMask, 14
qlaib::acquisition::BackendConfig, 15
coincidenceWindowPs, 15
exposureSeconds, 15
replayFile, 16
timestampBufferSize, 16
useMock, 16
qlaib::acquisition::BinReplayBackend, 16
coincWindowPs_, 19
currentSecond_, 19
lastSecond_, 19
nextBatch, 17
singles_, 19
start, 17
stop, 19
qlaib::acquisition::IBackend, 24
~IBackend, 25
nextBatch, 25
start, 25
stop, 25
qlaib::acquisition::MockBackend, 32
exposure_, 34
nextBatch, 33
rng_, 34
running_, 34
start, 33
startTime_, 34
stop, 33
qlaib::acquisition::QuTAGBackend, 35
bufferSize_, 37
chBuffer_, 37
coincWindowPs_, 37
connected_, 38
exposureMs_, 38
nextBatch, 36
recording_, 38
start, 36
startRecording, 37
stop, 37
stopRecording, 37
tsBuffer_, 38
qlaib::core, 14
qlaib::core::EventBus< Payload >, 22
Callback, 23
mutex_, 23
nextToken_, 23
publish, 23
subscribe, 23
subscribers_, 24
qlaib::core::EventBus< Payload >::Subscriber, 40
cb, 40
token, 40
qlaib::data, 14
qlaib::data::Coincidence, 19
counts, 20
label, 20
qlaib::data::SampleBatch, 39
coincidences, 39
metrics, 39
singles, 39
timestamp, 39
timestamps_ps, 40
qlaib::metrics, 14
qlaib::metrics::DummyQBER, 20
compute, 21
name, 21
qlaib::metrics::DummyVisibility, 21
compute, 22
name, 22
qlaib::metrics::IMetric, 26
~IMetric, 26
compute, 26
name, 26
qlaib::metrics::Registry, 38
computeAll, 38
metrics_, 39
registerMetric, 38
qlaib::ui, 14
qlaib::ui::MainWindow, 27
addPair, 28
appendSample, 28
backend_, 30

calibrateAll, 29
calibrateSelected, 29
cfg_, 30
coincidenceWindowPs_, 31
computeHistogram, 29
exportCsv, 29
histogramWindowPs_, 31
latestBatch_, 31
loadConfig, 29
MainWindow, 28
metrics_, 31
mode_, 31
pairs_, 31
refreshPairList, 29
refreshPairsTable, 29
removeSelectedPair, 29
replayFile_, 31
resetPairs, 29
saveConfig, 29
setMode, 30
setReplayFile, 30
setupUi, 30
start, 30
stop, 30
t0_ms_, 31
tick, 30
timer_, 31
toggleRecording, 30
qlaib::ui::PairSpec, 34
chA, 35
chB, 35
delayPs, 35
label, 35
QLaibLib Architecture (C++/Qt), 1
QLaibLib Usage (C++ Qt GUI), 3

recording_
qlaib::acquisition::QuTAGBackend, 38
refreshPairList
qlaib::ui::MainWindow, 29
refreshPairsTable
qlaib::ui::MainWindow, 29
registerMetric
qlaib::metrics::Registry, 38
removeSelectedPair
qlaib::ui::MainWindow, 29
replayFile
qlaib::acquisition::BackendConfig, 16
replayFile_
qlaib::ui::MainWindow, 31
resetPairs
qlaib::ui::MainWindow, 29
rng_
qlaib::acquisition::MockBackend, 34
running_
qlaib::acquisition::MockBackend, 34

saveConfig
qlaib::ui::MainWindow, 29
setMode
qlaib::ui::MainWindow, 30
setReplayFile
qlaib::ui::MainWindow, 30
setupUi
qlaib::ui::MainWindow, 30
singles
qlaib::data::SampleBatch, 39
singles_
qlaib::acquisition::BinReplayBackend, 19
src/acquisition/BinReplayBackend.cpp, 55
src/acquisition/MockBackend.cpp, 56
src/acquisition/QuTAGBackend.cpp, 56
src/metrics/Registry.cpp, 57
src/ui/MainWindow.cpp, 57
start
qlaib::acquisition::BinReplayBackend, 17
qlaib::acquisition::IBackend, 25
qlaib::acquisition::MockBackend, 33
qlaib::acquisition::QuTAGBackend, 36
qlaib::ui::MainWindow, 30
startRecording
qlaib::acquisition::QuTAGBackend, 37
startTime_
qlaib::acquisition::MockBackend, 34
stop
qlaib::acquisition::BinReplayBackend, 19
qlaib::acquisition::IBackend, 25
qlaib::acquisition::MockBackend, 33
qlaib::acquisition::QuTAGBackend, 37
qlaib::ui::MainWindow, 30
stopRecording
qlaib::acquisition::QuTAGBackend, 37
subscribe
qlaib::core::EventBus< Payload >, 23
subscribers_
qlaib::core::EventBus< Payload >, 24

t0_ms_
qlaib::ui::MainWindow, 31
tick
qlaib::ui::MainWindow, 30
timer_
qlaib::ui::MainWindow, 31
timestamp
qlaib::data::SampleBatch, 39
timestampBufferSize
qlaib::acquisition::BackendConfig, 16
timestamps_ps
qlaib::data::SampleBatch, 40
toggleRecording
qlaib::ui::MainWindow, 30
token
qlaib::core::EventBus< Payload >::Subscriber, 40
tsBuffer_
qlaib::acquisition::QuTAGBackend, 38

useMock
qlaib::acquisition::BackendConfig, 16