libquentier

0.5.0

Generated by Doxygen 1.8.13

Contents

1 libquentier **1**

2 Hierarchical Index **3**

2.1 Class Hierarchy . . . 3

3 Class Index **5**

3.1 Class List . . . 5

4 Class Documentation **9**

4.1 quentier::Account Class Reference . . . 9

4.1.1 Detailed Description . . . 10

4.1.2 Member Function Documentation . . . 11

4.1.2.1 displayName() . . . 11

4.1.2.2 evernoteAccountType() . . . 11

4.1.2.3 evernoteHost() . . . 11

4.1.2.4 id() . . . 11

4.1.2.5 isEmpty() . . . 12

4.1.2.6 name() . . . 12

4.1.2.7 setDisplayName() . . . 12

4.1.2.8 shardId() . . . 12

4.1.2.9 type() . . . 12

4.2 quentier::ApplicationSettings Class Reference . . . 13

4.2.1 Detailed Description . . . 14

4.2.2 Constructor & Destructor Documentation . . . 14

4.2.2.1 ApplicationSettings() [1/3] . . . 14

4.2.2.2 ApplicationSettings() [2/3] . . . 14
4.2.2.3 ApplicationSettings() [3/3] . . . 15
4.2.2.4 ~ApplicationSettings() . . . 15
4.2.3 Member Function Documentation . . . 15
4.2.3.1 beginGroup() [1/2] . . . 15
4.2.3.2 beginGroup() [2/2] . . . 16
4.2.3.3 beginReadArray() [1/2] . . . 16
4.2.3.4 beginReadArray() [2/2] . . . 16
4.2.3.5 beginWriteArray() [1/2] . . . 17
4.2.3.6 beginWriteArray() [2/2] . . . 17
4.2.3.7 contains() [1/2] . . . 17
4.2.3.8 contains() [2/2] . . . 18
4.2.3.9 remove() [1/2] . . . 18
4.2.3.10 remove() [2/2] . . . 19
4.2.3.11 setValue() [1/2] . . . 19
4.2.3.12 setValue() [2/2] . . . 19
4.2.3.13 value() [1/2] . . . 20
4.2.3.14 value() [2/2] . . . 20
4.3 quentier::ApplicationSettingsInitializationException Class Reference . . . 20
4.3.1 Detailed Description . . . 22
4.4 quentier::ApplicationSettings::ArrayCloser Struct Reference . . . 22
4.4.1 Detailed Description . . . 22
4.5 quentier::AuthenticationManager Class Reference . . . 23
4.5.1 Detailed Description . . . 24
4.6 quentier::ResourceRecognitionIndexItem::BarcodeItem Struct Reference . . . 24
4.7 quentier::DatabaseLockedException Class Reference . . . 24
4.8 quentier::DatabaseLockFailedException Class Reference . . . 25
4.9 quentier::DatabaseOpeningException Class Reference . . . 26
4.10 quentier::DatabaseRequestException Class Reference . . . 27
4.10.1 Detailed Description . . . 29

4.11 quentier::DateTimePrint Class Reference . . . 29
4.11.1 Detailed Description . . . 29
4.11.2 Member Enumeration Documentation . . . 29
4.11.2.1 Option . . . 29
4.12 quentier::DecryptedTextManager Class Reference . . . 29
4.13 quentier::DefaultLocalStorageCacheExpiryChecker Class Reference . . . 30
4.13.1 Detailed Description . . . 31
4.13.2 Member Function Documentation . . . 31
4.13.2.1 checkLinkedNotebooks() . . . 32
4.13.2.2 checkNotebooks() . . . 32
4.13.2.3 checkNotes() . . . 32
4.13.2.4 checkResources() . . . 32
4.13.2.5 checkSavedSearches() . . . 33
4.13.2.6 checkTags() . . . 33
4.13.2.7 clone() . . . 33
4.14 quentier::EmptyDataElementException Class Reference . . . 34
4.15 quentier::EncryptionManager Class Reference . . . 35
4.15.1 Detailed Description . . . 36
4.16 quentier::ENMLConverter Class Reference . . . 36
4.16.1 Detailed Description . . . 37
4.16.2 Member Function Documentation . . . 37
4.16.2.1 cleanupExternalHtml() . . . 37
4.16.2.2 exportNotesToEnex() . . . 38
4.16.2.3 htmlToQTextDocument() . . . 38
4.16.2.4 importEnex() . . . 39
4.17 quentier::ErrorString Class Reference . . . 39
4.17.1 Detailed Description . . . 41
4.18 quentier::EventLoopWithExitStatus Class Reference . . . 41
4.19 quentier::FileCopier Class Reference . . . 42
4.20 quentier::FileIOProcessorAsync Class Reference . . . 44

4.20.1 Detailed Description . . . 45
4.20.2 Member Function Documentation . . . 45
4.20.2.1 onReadFileRequest . . . 45
4.20.2.2 onWriteFileRequest . . . 45
4.20.2.3 readFileRequestProcessed . . . 46
4.20.2.4 setIdleTimePeriod() . . . 46
4.20.2.5 writeFileRequestProcessed . . . 47
4.21 quentier::FileSystemWatcher Class Reference . . . 47
4.22 quentier::ApplicationSettings::GroupCloser Struct Reference . . . 48
4.22.1 Detailed Description . . . 49
4.23 quentier::HTMLCleaner Class Reference . . . 49
4.24 quentier::IAuthenticationManager Class Reference . . . 49
4.25 quentier::IFavoritableDataElement Class Reference . . . 50
4.25.1 Detailed Description . . . 51
4.26 quentier::IKeychainService Class Reference . . . 51
4.26.1 Detailed Description . . . 53
4.26.2 Member Enumeration Documentation . . . 53
4.26.2.1 ErrorCode . . . 53
4.26.3 Member Function Documentation . . . 53
4.26.3.1 deletePasswordJobFinished . . . 53
4.26.3.2 readPasswordJobFinished . . . 55
4.26.3.3 startDeletePasswordJob() . . . 55
4.26.3.4 startReadPasswordJob() . . . 56
4.26.3.5 startWritePasswordJob() . . . 56
4.26.3.6 writePasswordJobFinished . . . 56
4.27 quentier::ILocalStorageCacheExpiryChecker Class Reference . . . 57
4.27.1 Detailed Description . . . 58
4.27.2 Member Function Documentation . . . 59
4.27.2.1 checkLinkedNotebooks() . . . 59
4.27.2.2 checkNotebooks() . . . 59

4.27.2.3 checkNotes() . . . 59
4.27.2.4 checkResources() . . . 59
4.27.2.5 checkSavedSearches() . . . 60
4.27.2.6 checkTags() . . . 60
4.27.2.7 clone() . . . 60
4.28 quentier::ILocalStorageDataElement Class Reference . . . 60
4.29 quentier::ILocalStoragePatch Class Reference . . . 61
4.29.1 Detailed Description . . . 62
4.29.2 Member Function Documentation . . . 62
4.29.2.1 apply() . . . 62
4.29.2.2 backupLocalStorage() . . . 63
4.29.2.3 backupProgress . . . 63
4.29.2.4 fromVersion() . . . 63
4.29.2.5 patchLongDescription() . . . 63
4.29.2.6 patchShortDescription() . . . 64
4.29.2.7 progress . . . 64
4.29.2.8 removeLocalStorageBackup() . . . 64
4.29.2.9 restoreBackupProgress . . . 64
4.29.2.10 restoreLocalStorageFromBackup() . . . 65
4.29.2.11 toVersion() . . . 65
4.30 quentier::INoteEditorBackend Class Reference . . . 66
4.31 quentier::INoteStore Class Reference . . . 69
4.31.1 Detailed Description . . . 70
4.31.2 Member Function Documentation . . . 71
4.31.2.1 authenticateToSharedNotebook() . . . 71
4.31.2.2 createNote() . . . 71
4.31.2.3 createNotebook() . . . 72
4.31.2.4 createSavedSearch() . . . 72
4.31.2.5 createTag() . . . 73
4.31.2.6 getLinkedNotebookSyncChunk() . . . 73

4.31.2.7 getLinkedNotebookSyncState() . . . 74
4.31.2.8 getNote() . . . 75
4.31.2.9 getNoteAsync() . . . 75
4.31.2.10 getResource() . . . 76
4.31.2.11 getResourceAsync() . . . 77
4.31.2.12 getSyncChunk() . . . 78
4.31.2.13 getSyncState() . . . 78
4.31.2.14 noteStoreUrl() . . . 79
4.31.2.15 setAuthData() . . . 79
4.31.2.16 setNoteStoreUrl() . . . 79
4.31.2.17 stop() . . . 79
4.31.2.18 updateNote() . . . 79
4.31.2.19 updateNotebook() . . . 80
4.31.2.20 updateSavedSearch() . . . 80
4.31.2.21 updateTag() . . . 81
4.32 quentier::INoteStoreDataElement Class Reference . . . 81
4.33 quentier::IQuentierException Class Reference . . . 82
4.33.1 Detailed Description . . . 84
4.34 quentier::ISyncChunksDataCounters Struct Reference . . . 84
4.34.1 Detailed Description . . . 85
4.34.2 Member Function Documentation . . . 85
4.34.2.1 addedLinkedNotebooks() . . . 85
4.34.2.2 addedNotebooks() . . . 85
4.34.2.3 addedSavedSearches() . . . 86
4.34.2.4 addedTags() . . . 86
4.34.2.5 expungedLinkedNotebooks() . . . 86
4.34.2.6 expungedNotebooks() . . . 86
4.34.2.7 expungedSavedSearches() . . . 86
4.34.2.8 expungedTags() . . . 86
4.34.2.9 totalExpungedLinkedNotebooks() . . . 86

4.34.2.10 totalExpungedNotebooks() . . . 87
4.34.2.11 totalExpungedSavedSearches() . . . 87
4.34.2.12 totalExpungedTags() . . . 87
4.34.2.13 totalLinkedNotebooks() . . . 87
4.34.2.14 totalNotebooks() . . . 87
4.34.2.15 totalSavedSearches() . . . 87
4.34.2.16 totalTags() . . . 87
4.34.2.17 updatedLinkedNotebooks() . . . 88
4.34.2.18 updatedNotebooks() . . . 88
4.34.2.19 updatedSavedSearches() . . . 88
4.34.2.20 updatedTags() . . . 88
4.35 quentier::ISyncStateStorage::ISyncState Class Reference . . . 88
4.35.1 Detailed Description . . . 89
4.36 quentier::ISyncStateStorage Class Reference . . . 89
4.36.1 Detailed Description . . . 90
4.36.2 Member Function Documentation . . . 90
4.36.2.1 notifySyncStateUpdated . . . 91
4.37 quentier::IUserStore Class Reference . . . 91
4.37.1 Detailed Description . . . 91
4.37.2 Member Function Documentation . . . 91
4.37.2.1 checkVersion() . . . 91
4.37.2.2 getAccountLimits() . . . 92
4.37.2.3 getUser() . . . 92
4.37.2.4 setAuthData() . . . 93
4.38 quentier::LimitedStack< T > Class Template Reference . . . 93
4.38.1 Detailed Description . . . 94
4.39 quentier::LinkedNotebook Class Reference . . . 94
4.40 quentier::LocalStorageCacheManager Class Reference . . . 96
4.41 quentier::LocalStorageCacheManagerException Class Reference . . . 98
4.42 quentier::LocalStorageManager Class Reference . . . 99

4.42.1 Member Enumeration Documentation . . . 105
4.42.1.1 GetNoteOption . . . 105
4.42.1.2 GetResourceOption . . . 106
4.42.1.3 ListObjectsOption . . . 106
4.42.1.4 StartupOption . . . 106
4.42.1.5 UpdateNoteOption . . . 107
4.42.2 Constructor & Destructor Documentation . . . 107
4.42.2.1 LocalStorageManager() . . . 107
4.42.3 Member Function Documentation . . . 107
4.42.3.1 accountHighUsn() . . . 108
4.42.3.2 addEnResource() . . . 108
4.42.3.3 addLinkedNotebook() . . . 108
4.42.3.4 addNote() . . . 109
4.42.3.5 addNotebook() . . . 109
4.42.3.6 addSavedSearch() . . . 110
4.42.3.7 addTag() . . . 110
4.42.3.8 addUser() . . . 111
4.42.3.9 deleteUser() . . . 111
4.42.3.10 enResourceCount() . . . 111
4.42.3.11 expungeEnResource() . . . 112
4.42.3.12 expungeLinkedNotebook() . . . 112
4.42.3.13 expungeNote() . . . 113
4.42.3.14 expungeNotebook() . . . 113
4.42.3.15 expungeNotelessTagsFromLinkedNotebooks() . . . 114
4.42.3.16 expungeSavedSearch() . . . 114
4.42.3.17 expungeTag() . . . 114
4.42.3.18 expungeUser() . . . 115
4.42.3.19 findDefaultNotebook() . . . 115
4.42.3.20 findDefaultOrLastUsedNotebook() . . . 116
4.42.3.21 findEnResource() . . . 116

4.42.3.22 findLastUsedNotebook() . . . 117
4.42.3.23 findLinkedNotebook() . . . 117
4.42.3.24 findNote() . . . 118
4.42.3.25 findNotebook() . . . 118
4.42.3.26 findNoteLocalUidsWithSearchQuery() . . . 119
4.42.3.27 findNotesWithSearchQuery() . . . 119
4.42.3.28 findSavedSearch() . . . 119
4.42.3.29 findTag() . . . 120
4.42.3.30 findUser() . . . 120
4.42.3.31 highestSupportedLocalStorageVersion() . . . 121
4.42.3.32 isLocalStorageVersionTooHigh() . . . 121
4.42.3.33 linkedNotebookCount() . . . 122
4.42.3.34 listAllLinkedNotebooks() . . . 122
4.42.3.35 listAllNotebooks() . . . 122
4.42.3.36 listAllSavedSearches() . . . 123
4.42.3.37 listAllSharedNotebooks() . . . 124
4.42.3.38 listAllTags() . . . 124
4.42.3.39 listAllTagsPerNote() . . . 125
4.42.3.40 listLinkedNotebooks() . . . 125
4.42.3.41 listNotebooks() . . . 126
4.42.3.42 listNotes() . . . 127
4.42.3.43 listNotesByLocalUids() . . . 127
4.42.3.44 listNotesPerNotebook() . . . 128
4.42.3.45 listNotesPerNotebooksAndTags() . . . 129
4.42.3.46 listNotesPerTag() . . . 130
4.42.3.47 listSavedSearches() . . . 130
4.42.3.48 listSharedNotebooksPerNotebookGuid() . . . 131
4.42.3.49 listTags() . . . 131
4.42.3.50 listTagsWithNoteLocalUids() . . . 132
4.42.3.51 localStorageRequiresUpgrade() . . . 133

4.42.3.52 localStorageVersion() . . . 133
4.42.3.53 notebookCount() . . . 134
4.42.3.54 noteCount() . . . 134
4.42.3.55 noteCountPerNotebook() . . . 134
4.42.3.56 noteCountPerNotebooksAndTags() . . . 135
4.42.3.57 noteCountPerTag() . . . 135
4.42.3.58 noteCountsPerAllTags() . . . 136
4.42.3.59 requiredLocalStoragePatches() . . . 136
4.42.3.60 savedSearchCount() . . . 137
4.42.3.61 switchUser() . . . 137
4.42.3.62 tagCount() . . . 137
4.42.3.63 updateEnResource() . . . 138
4.42.3.64 updateLinkedNotebook() . . . 138
4.42.3.65 updateNote() . . . 139
4.42.3.66 updateNotebook() . . . 139
4.42.3.67 updateSavedSearch() . . . 140
4.42.3.68 updateTag() . . . 140
4.42.3.69 updateUser() . . . 141
4.42.3.70 upgradeProgress . . . 141
4.42.3.71 userCount() . . . 142
4.43 quentier::LocalStorageManagerAsync Class Reference . . . 142
4.44 quentier::LoggerInitializationException Class Reference . . . 149
4.45 quentier::LRUCache< Key, Value, Allocator > Class Template Reference . . . 150
4.46 quentier::Note Class Reference . . . 151
4.47 quentier::Notebook Class Reference . . . 154
4.48 quentier::ENMLConverter::NoteContentToHtmlExtraData Struct Reference . . . 158
4.49 quentier::NoteEditor Class Reference . . . 158
4.49.1 Detailed Description . . . 161
4.49.2 Member Function Documentation . . . 161
4.49.2.1 backend() . . . 162

4.49.2.2 clear() . . . 162
4.49.2.3 convertToNote . . . 162
4.49.2.4 currentNoteLocalUid() . . . 162
4.49.2.5 defaultFont() . . . 162
4.49.2.6 defaultPalette() . . . 162
4.49.2.7 idleTime() . . . 163
4.49.2.8 inAppNoteLinkPasteRequested . . . 163
4.49.2.9 initialize() . . . 163
4.49.2.10 isEditorPageModified() . . . 164
4.49.2.11 isModified() . . . 164
4.49.2.12 isNoteLoaded() . . . 164
4.49.2.13 saveNoteToLocalStorage . . . 164
4.49.2.14 setAccount() . . . 164
4.49.2.15 setBackend() . . . 164
4.49.2.16 setCurrentNoteLocalUid() . . . 164
4.49.2.17 setDefaultFont . . . 165
4.49.2.18 setDefaultPalette . . . 165
4.49.2.19 setFocus() . . . 165
4.49.2.20 setInitialPageHtml() . . . 166
4.49.2.21 setNoteDeletedPageHtml() . . . 166
4.49.2.22 setNoteLoadingPageHtml() . . . 166
4.49.2.23 setNoteNotFoundPageHtml() . . . 166
4.49.2.24 setNoteTitle . . . 166
4.49.2.25 setTagIds . . . 166
4.49.2.26 setUndoStack() . . . 167
4.49.2.27 undoStack() . . . 167
4.50 quentier::NoteEditorInitializationException Class Reference . . . 167
4.51 quentier::NoteEditorPluginInitializationException Class Reference . . . 168
4.52 quentier::NoteSearchQuery Class Reference . . . 169
4.52.1 Member Function Documentation . . . 172

4.52.1.1 notebookModifier() . . . 172
4.52.1.2 queryString() . . . 172
4.53 quentier::NullPtrException Class Reference . . . 172
4.54 quentier::ResourceRecognitionIndexItem::ObjectItem Struct Reference . . . 173
4.55 quentier::Printable Class Reference . . . 174
4.55.1 Detailed Description . . . 175
4.56 quentier::QuentierApplication Class Reference . . . 175
4.57 quentier::QuentierUndoCommand Class Reference . . . 176
4.57.1 Detailed Description . . . 177
4.58 quentier::Resource Class Reference . . . 177
4.59 quentier::ResourceRecognitionIndexItem Class Reference . . . 180
4.60 quentier::ResourceRecognitionIndices Class Reference . . . 182
4.61 quentier::SavedSearch Class Reference . . . 183
4.62 quentier::ResourceRecognitionIndexItem::ShapeItem Struct Reference . . . 185
4.63 quentier::SharedNote Class Reference . . . 186
4.64 quentier::SharedNotebook Class Reference . . . 188
4.65 quentier::ShortcutManager Class Reference . . . 190
4.65.1 Member Function Documentation . . . 191
4.65.1.1 defaultShortcut() [1/2] . . . 192
4.65.1.2 defaultShortcut() [2/2] . . . 192
4.65.1.3 shortcut() [1/2] . . . 192
4.65.1.4 shortcut() [2/2] . . . 192
4.65.1.5 userShortcut() [1/2] . . . 193
4.65.1.6 userShortcut() [2/2] . . . 193
4.66 quentier::ENMLConverter::SkipHtmlElementRule Class Reference . . . 193
4.66.1 Detailed Description . . . 194
4.67 quentier::SpellChecker Class Reference . . . 195
4.68 quentier::StringUtils::StringFilterPredicate Struct Reference . . . 196
4.69 quentier::StringUtils Class Reference . . . 196
4.70 quentier::SynchronizationManager Class Reference . . . 196

4.70.1 Detailed Description . . . 198
4.70.2 Constructor & Destructor Documentation . . . 198
4.70.2.1 SynchronizationManager() . . . 198
4.70.3 Member Function Documentation . . . 199
4.70.3.1 active() . . . 199
4.70.3.2 authenticate . . . 199
4.70.3.3 authenticateCurrentAccount . . . 199
4.70.3.4 authenticationFinished . . . 200
4.70.3.5 authenticationRevoked . . . 200
4.70.3.6 detectedConflictDuringLocalChangesSending . . . 200
4.70.3.7 downloadNoteThumbnailsOption() . . . 201
4.70.3.8 failed . . . 201
4.70.3.9 finished . . . 201
4.70.3.10 linkedNotebooksNotesDownloadProgress . . . 201
4.70.3.11 linkedNotebooksResourcesDownloadProgress . . . 202
4.70.3.12 linkedNotebooksSyncChunksDownloaded . . . 202
4.70.3.13 linkedNotebookSyncChunksDataProcessingProgress . . . 202
4.70.3.14 linkedNotebookSyncChunksDownloadProgress . . . 202
4.70.3.15 notesDownloadProgress . . . 203
4.70.3.16 preparedDirtyObjectsForSending . . . 203
4.70.3.17 preparedLinkedNotebooksDirtyObjectsForSending . . . 203
4.70.3.18 rateLimitExceeded . . . 203
4.70.3.19 remoteToLocalSyncDone . . . 204
4.70.3.20 remoteToLocalSyncStopped . . . 204
4.70.3.21 resourcesDownloadProgress . . . 204
4.70.3.22 revokeAuthentication . . . 204
4.70.3.23 sendLocalChangesStopped . . . 205
4.70.3.24 setAccount . . . 205
4.70.3.25 setAccountDone . . . 205
4.70.3.26 setDownloadInkNoteImages . . . 205

4.70.3.27 setDownloadInkNoteImagesDone . . . 205
4.70.3.28 setDownloadNoteThumbnails . . . 206
4.70.3.29 setDownloadNoteThumbnailsDone . . . 206
4.70.3.30 setInkNoteImagesStoragePath . . . 206
4.70.3.31 setInkNoteImagesStoragePathDone . . . 206
4.70.3.32 started . . . 206
4.70.3.33 stop . . . 207
4.70.3.34 stopped . . . 207
4.70.3.35 syncChunksDataProcessingProgress . . . 207
4.70.3.36 syncChunksDownloaded . . . 207
4.70.3.37 syncChunksDownloadProgress . . . 207
4.70.3.38 synchronize . . . 208
4.70.3.39 willRepeatRemoteToLocalSyncAfterSendingChanges . . . 208
4.71 quentier::SysInfo Class Reference . . . 208
4.72 quentier::Tag Class Reference . . . 208
4.73 quentier::ResourceRecognitionIndexItem::TextItem Struct Reference . . . 210
4.74 quentier::UidGenerator Class Reference . . . 210
4.75 quentier::User Class Reference . . . 210

Index **213**

Chapter 1

libquentier

Set of Qt/C++ APIs for feature rich desktop clients for Evernote service

What's this

This library presents a set of Qt/C++ APIs useful for applications working as feature rich desktop clients for Evernote service. The most important and useful components of the library are the following:

- Local storage - persistence of data downloaded from Evernote service in a local SQLite database
- Synchronization - the logics of exchanging new and/or modified data with Evernote service
- Note editor - the UI component capable for notes displaying and editing

The library is based on the lower level functionality provided by `QEverCloud` library. It also serves as the functional core of `Quentier` application.

WARNING: libquentier is in alpha state right now, neither API nor ABI can be considered stable yet!

How to build/install

Please see the building/installation guide.

How to contribute

Please see the contribution guide for detailed info.

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

quentier::ApplicationSettings::ArrayCloser . . . 22
quentier::ResourceRecognitionIndexItem::BarcodeItem . . . 24
quentier::DateTimePrint . . . 29
quentier::DecryptedTextManager . . . 29
quentier::ENMLConverter . . . 36
exception
  quentier::IQuentierException . . . 82
    quentier::ApplicationSettingsInitializationException . . . 20
    quentier::DatabaseLockedException . . . 24
    quentier::DatabaseLockFailedException . . . 25
    quentier::DatabaseOpeningException . . . 26
    quentier::DatabaseRequestException . . . 27
    quentier::EmptyDataElementException . . . 34
    quentier::LocalStorageCacheManagerException . . . 98
    quentier::LoggerInitializationException . . . 149
    quentier::NoteEditorInitializationException . . . 167
    quentier::NoteEditorPluginInitializationException . . . 168
    quentier::NullPtrException . . . 172
quentier::ApplicationSettings::GroupCloser . . . 48
quentier::HTMLCleaner . . . 49
quentier::ILocalStorageDataElement . . . 60
  quentier::INoteStoreDataElement . . . 81
    quentier::IFavoritableDataElement . . . 50
      quentier::Note . . . 151
      quentier::Notebook . . . 154
      quentier::SavedSearch . . . 183
      quentier::Tag . . . 208
    quentier::LinkedNotebook . . . 94
    quentier::Resource . . . 177
quentier::INoteEditorBackend . . . 66
quentier::IUserStore . . . 91
quentier::LRUCache< Key, Value, Allocator > . . . 150
quentier::ENMLConverter::NoteContentToHtmlExtraData . . . 158
quentier::ResourceRecognitionIndexItem::ObjectItem . . . 173
quentier::Printable . . . 174

quentier::Account . . . 9
quentier::ApplicationSettings . . . 13
quentier::ENMLConverter::SkipHtmlElementRule . . . 193
quentier::ErrorString . . . 39
quentier::ILocalStorageCacheExpiryChecker . . . 57
quentier::DefaultLocalStorageCacheExpiryChecker . . . 30
quentier::INoteStoreDataElement . . . 81
quentier::IQuentierException . . . 82
quentier::ISyncChunksDataCounters . . . 84
quentier::ISyncStateStorage::ISyncState . . . 88
quentier::LocalStorageCacheManager . . . 96
quentier::NoteSearchQuery . . . 169
quentier::ResourceRecognitionIndexItem . . . 180
quentier::ResourceRecognitionIndices . . . 182
quentier::SharedNote . . . 186
quentier::SharedNotebook . . . 188
quentier::User . . . 210

QApplication
quentier::QuentierApplication . . . 175

QEventLoop
quentier::EventLoopWithExitStatus . . . 41

QObject
quentier::EncryptionManager . . . 35
quentier::FileCopier . . . 42
quentier::FileIOProcessorAsync . . . 44
quentier::FileSystemWatcher . . . 47
quentier::IAuthenticationManager . . . 49
quentier::AuthenticationManager . . . 23
quentier::IKeychainService . . . 51
quentier::ILocalStoragePatch . . . 61
quentier::INoteStore . . . 69
quentier::ISyncStateStorage . . . 89
quentier::LocalStorageManager . . . 99
quentier::LocalStorageManagerAsync . . . 142
quentier::QuentierUndoCommand . . . 176
quentier::ShortcutManager . . . 190
quentier::SpellChecker . . . 195
quentier::SynchronizationManager . . . 196

QSettings
quentier::ApplicationSettings . . . 13

QStack
quentier::LimitedStack< T > . . . 93

QUndoCommand
quentier::QuentierUndoCommand . . . 176

QWidget
quentier::NoteEditor . . . 158

quentier::ResourceRecognitionIndexItem::ShapeItem . . . 185
quentier::StringUtils::StringFilterPredicate . . . 196
quentier::StringUtils . . . 196
quentier::SysInfo . . . 208
quentier::ResourceRecognitionIndexItem::TextItem . . . 210
quentier::UidGenerator . . . 210

Chapter 3

Class Index

3.1 Class List

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

quentier::Account
: Encapsulates some details about the account: its name, whether it is local or synchronized to Evernote and for the latter case - some additional details like upload limit etc . . . 9

quentier::ApplicationSettings
: Enhances the functionality of QSettings, in particular it simplifies the way of working with either application-wide or account-specific settings . . . 13

quentier::ApplicationSettingsInitializationException
: The ApplicationSettingsInitializationException can be thrown from methods of ApplicationSettings class if it's unable to locate the file with persistent settings . . . 20

quentier::ApplicationSettings::ArrayCloser . . . 22

quentier::AuthenticationManager
: Libquentier's default implementation of IAuthenticationManager interface; internally uses QEverCloud's OAuth widget . . . 23

quentier::ResourceRecognitionIndexItem::BarcodeItem . . . 24

quentier::DatabaseLockedException . . . 24

quentier::DatabaseLockFailedException . . . 25

quentier::DatabaseOpeningException . . . 26

quentier::DatabaseRequestException
: The DatabaseRequestException is thrown when the local storage database encounters some internal error during the attemt to serve a request . . . 27

quentier::DateTimePrint
: Simply wraps the enum containing datetime printing options . . . 29

quentier::DecryptedTextManager . . . 29

quentier::DefaultLocalStorageCacheExpiryChecker . . . 30

quentier::EmptyDataElementException . . . 34

quentier::EncryptionManager
: Both synchronous methods to encrypt or decrypt given text with password, cipher and key length and their signal-slot based potentially asynchronous counterparts . . . 35

quentier::ENMLConverter
: Encapsulates a set of methods and helper data structures for performing the conversions between ENML and other note content formats, namely HTML . . . 36

quentier::ErrorString
: Encapsulates two (or more) strings which are meant to contain translatable (base) and non-translatable (details) parts of the error description . . . 39

quentier::EventLoopWithExitStatus . . . 41

quentier::FileCopier . . . 42
quentier::FileIOProcessorAsync
Wrapper under simple file IO operations, it is meant to be used for simple asynchronous IO . . . 44
quentier::FileSystemWatcher . . . 47
quentier::ApplicationSettings::GroupCloser . . . 48
quentier::HTMLCleaner . . . 49
quentier::IAuthenticationManager . . . 49
quentier::IFavoritableDataElement . . . 50
quentier::IKeychainService
The IKeychainService interface provides methods intended to start potentially asynchronous interaction with the keychain and signals intended to notify listeners about the completion of asynchronous interactions . . . 51
quentier::ILocalStorageCacheExpiryChecker
Interface for cache expiry checker used by LocalStorageCacheManager to see whether particular caches (of notes, notebooks, tags, linked notebooks and/or saved searches) need to be shrunk 57
quentier::ILocalStorageDataElement . . . 60
quentier::ILocalStoragePatch
Interface for patches of local storage. Each such patch somehow changes the layout of local storage persistence so that only compliant & corresponding versions of libquentier can be used to work with it . . . 61
quentier::INoteEditorBackend . . . 66
quentier::INoteStore
INoteStore is the interface which provides methods required for the implementation of NoteStore part of Evernote EDAM sync protocol . . . 69
quentier::INoteStoreDataElement . . . 81
quentier::IQuentierException
Interface for exceptions specific to libquentier and applications based on it . . . 82
quentier::ISyncChunksDataCounters
The ISyncChunksDataCounters interface provides integer counters representing the current progress on processing the data from downloaded sync chunks . . . 84
quentier::ISyncStateStorage::ISyncState
The ISyncState interface provides accessory methods to determine the sync state for the account 88
quentier::ISyncStateStorage
The ISyncStateStorage interface represents the interface of a class which stores sync state for given accounts persistently and provides access to previously stores sync states . . . 89
quentier::IUserStore
IUserStore is the interface which provides methods required for the implementation of UserStore part of Evernote EDAM sync protocol . . . 91
quentier::LimitedStack< T >
The LimitedStack template class implements a stack which may have a limitation for its size; when the size becomes too much according to the limit, the bottom element of the stack gets erased from it. Only limits greater than zero are considered . . . 93
quentier::LinkedNotebook . . . 94
quentier::LocalStorageCacheManager . . . 96
quentier::LocalStorageCacheManagerException . . . 98
quentier::LocalStorageManager . . . 99
quentier::LocalStorageManagerAsync . . . 142
quentier::LoggerInitializationException . . . 149
quentier::LRUCache< Key, Value, Allocator > . . . 150
quentier::Note . . . 151
quentier::Notebook . . . 154
quentier::ENMLConverter::NoteContentToHtmlExtraData . . . 158
quentier::NoteEditor
Widget encapsulating all the functionality necessary for showing and editing notes . . . 158
quentier::NoteEditorInitializationException . . . 167
quentier::NoteEditorPluginInitializationException . . . 168
quentier::NoteSearchQuery . . . 169
quentier::NullPtrException . . . 172

quentier::ResourceRecognitionIndexItem::ObjectItem . . . 173
quentier::Printable
Interface for Quentier's internal classes which should be able to write themselves into QTextStream and/or convert to QString . . . 174
quentier::QuentierApplication . . . 175
quentier::QuentierUndoCommand
Has the sole purpose of working around one quirky aspect of Qt's undo/redo framework: when you push QUndoCommand to QUndoStack, it calls "redo" method of that command. This class offers subclasses to implement their own methods for actual "undo" and "redo" commands while ignoring the attempts to "redo" anything if there were no previous "undo" call prior to that . . . 176
quentier::Resource . . . 177
quentier::ResourceRecognitionIndexItem . . . 180
quentier::ResourceRecognitionIndices . . . 182
quentier::SavedSearch . . . 183
quentier::ResourceRecognitionIndexItem::ShapeItem . . . 185
quentier::SharedNote . . . 186
quentier::SharedNotebook . . . 188
quentier::ShortcutManager . . . 190
quentier::ENMLConverter::SkipHtmlElementRule
Describes the set of rules for HTML -> ENML conversion about the HTML elements that should not be actually converted to ENML due to their nature of being "helper" elements for the display or functioning of something within the note editor's page. The HTML -> ENML conversion would ignore tags and attributes forbidden by ENML even without these rules conditionally preserving or skipping the contents and nested elements of skipped elements . . . 193
quentier::SpellChecker . . . 195
quentier::StringUtils::StringFilterPredicate . . . 196
quentier::StringUtils . . . 196
quentier::SynchronizationManager
Encapsulates methods and signals & slots required to perform the full or partial synchronization of data with remote Evernote servers. The class also deals with authentication with Evernote service through OAuth . . . 196
quentier::SysInfo . . . 208
quentier::Tag . . . 208
quentier::ResourceRecognitionIndexItem::TextItem . . . 210
quentier::UidGenerator . . . 210
quentier::User . . . 210

Chapter 4

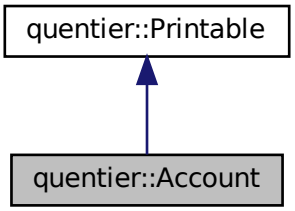
Class Documentation

4.1 quentier::Account Class Reference

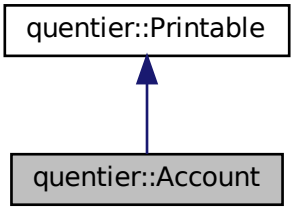
The Account class encapsulates some details about the account: its name, whether it is local or synchronized to Evernote and for the latter case - some additional details like upload limit etc.

`#include <Account.h>`

Inheritance diagram for quentier::Account:



Collaboration diagram for quentier::Account:



Public Types

- enum **Type** { **Local** = 0, **Evernote** }
- enum **EvernoteAccountType** { **Free** = 0, **Plus**, **Premium**, **Business** }

Public Member Functions

- **Account** (QString name, const Type type, const qevercloud::UserID userId=-1, const EvernoteAccountType evernoteAccountType=EvernoteAccountType::Free, QString evernoteHost={}, QString shardId={})
- **Account** (const Account &other)
- Account & **operator=** (const Account &other)
- bool **operator==** (const Account &other) const
- bool **operator!=** (const Account &other) const
- bool isEmpty () const
- QString name () const
- void setName (QString name)

  setName sets the username to the account
- QString displayName () const
- void setDisplayName (QString displayName)
- Type type () const
- qevercloud::UserID id () const
- EvernoteAccountType evernoteAccountType () const
- QString evernoteHost () const
- QString shardId () const
- void **setEvernoteAccountType** (const EvernoteAccountType evernoteAccountType)
- void **setEvernoteHost** (QString evernoteHost)
- void **setShardId** (QString shardId)
- qint32 **mailLimitDaily** () const
- qint64 **noteSizeMax** () const
- qint64 **resourceSizeMax** () const
- qint32 **linkedNotebookMax** () const
- qint32 **noteCountMax** () const
- qint32 **notebookCountMax** () const
- qint32 **tagCountMax** () const
- qint32 **noteTagCountMax** () const
- qint32 **savedSearchCountMax** () const
- qint32 **noteResourceCountMax** () const
- void **setEvernoteAccountLimits** (const qevercloud::AccountLimits &limits)
- virtual QTextStream & **print** (QTextStream &strm) const override

Friends

- QUENTIER_EXPORT QTextStream & **operator<<** (QTextStream &strm, const Type type)
- QUENTIER_EXPORT QDebug & **operator<<** (QDebug &dbg, const Type type)
- QUENTIER_EXPORT QTextStream & **operator<<** (QTextStream &strm, const EvernoteAccountType type)
- QUENTIER_EXPORT QDebug & **operator<<** (QDebug &dbg, const EvernoteAccountType type)

Additional Inherited Members

4.1.1 Detailed Description

The Account class encapsulates some details about the account: its name, whether it is local or synchronized to Evernote and for the latter case - some additional details like upload limit etc.

4.1.2 Member Function Documentation

4.1.2.1 displayName()

```
QString quentier::Account::displayName ( ) const
```

Returns

Printable user's name that is not used to uniquely identify the account, so this name may repeat across different local and Evernote accounts

4.1.2.2 evernoteAccountType()

```
EvernoteAccountType quentier::Account::evernoteAccountType ( ) const
```

Returns

The type of the Evernote account; if applied to free account, returns "Free"

4.1.2.3 evernoteHost()

```
QString quentier::Account::evernoteHost ( ) const
```

Returns

The Evernote server host with which the account is associated

4.1.2.4 id()

```
qevercloud::UserID quentier::Account::id ( ) const
```

Returns

User id for Evernote accounts, -1 for local accounts (as the concept of user id is not defined for local accounts)

4.1.2.5 isEmpty()

```
bool quentier::Account::isEmpty ( ) const
```

Returns

True if either the account is local but the name is empty or if the account is Evernote but user id is negative; in all other cases return false

4.1.2.6 name()

```
QString quentier::Account::name ( ) const
```

Returns

Username for either local or Evernote account

4.1.2.7 setDisplayName()

```
void quentier::Account::setDisplayName (
            QString displayName )
```

Set the printable name of the account

4.1.2.8 shardId()

```
QString quentier::Account::shardId ( ) const
```

Returns

Shard id for Evernote accounts, empty string for local accounts (as the concept of shard id is not defined for local accounts)

4.1.2.9 type()

```
Type quentier::Account::type ( ) const
```

Returns

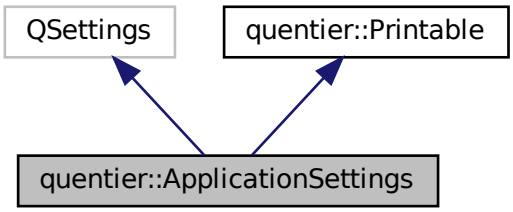
The type of the account: either local of Evernote

4.2 quentier::ApplicationSettings Class Reference

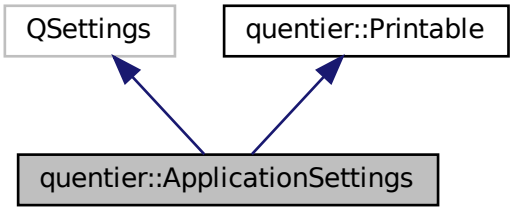
The ApplicationSettings class enhances the functionality of QSettings, in particular it simplifies the way of working with either application-wide or account-specific settings.

`#include <ApplicationSettings.h>`

Inheritance diagram for quentier::ApplicationSettings:



Collaboration diagram for quentier::ApplicationSettings:



Classes

- struct ArrayCloser
- struct GroupCloser

Public Member Functions

- ApplicationSettings (const QString &settingsName={})
- ApplicationSettings (const Account &account, const QString &settingsName={})
- ApplicationSettings (const Account &account, const char *settingsName, const int settingsNameSize=-1)
- virtual ~ApplicationSettings () override
- void beginGroup (const QString &prefix)

- void beginGroup (const char *prefix, const int size=-1)
- int beginReadArray (const QString &prefix)
- int beginReadArray (const char *prefix, const int size=-1)
- void beginWriteArray (const QString &prefix, const int arraySize=-1)
- void beginWriteArray (const char *prefix, const int arraySize=-1, const int prefixSize=-1)
- bool contains (const QString &key) const
- bool contains (const char *key, const int size=-1) const
- void remove (const QString &key)
- void remove (const char *key, const int size=-1)
- void setValue (const QString &key, const QVariant &value)
- void setValue (const char *key, const QVariant &value, const int keySize=-1)
- QVariant value (const QString &key, const QVariant &defaultValue={}) const
- QVariant value (const char *key, const QVariant &defaultValue={}, const int keySize=-1) const
- virtual QTextStream & **print** (QTextStream &strm) const override

Additional Inherited Members

4.2.1 Detailed Description

The ApplicationSettings class enhances the functionality of QSettings, in particular it simplifies the way of working with either application-wide or account-specific settings.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 ApplicationSettings() [1/3]

```
quentier::ApplicationSettings::ApplicationSettings (
            const QString & settingsName = {} )
```

Constructor for application settings not being account-specific

Parameters

settingsName	If not empty, the created application settings would manage the settings stored in a file with a specific name within the common settings storage; otherwise they would be stored in the default settings file for the account

4.2.2.2 ApplicationSettings() [2/3]

```
quentier::ApplicationSettings::ApplicationSettings (
            const Account & account,
            const QString & settingsName = {} )
```

Constructor for application settings specific to the account

Parameters

account	The account for which the settings are to be stored or read
settingsName	If not empty, the created application settings would manage the settings stored in a file with a specific name within the account's settings storage; otherwise they would be stored in the default settings file for the account

4.2.2.3 ApplicationSettings() [3/3]

```
quentier::ApplicationSettings::ApplicationSettings (
            const Account & account,
            const char * settingsName,
            const int settingsNameSize = -1 )
```

Constructor for application settings specific to the account

Parameters

account	The account for which the settings are to be stored or read
settingsName	If not nullptr, the created application settings would manage the settings stored in a file with a specific name within the account's settings storage; otherwise they would be stored in the default settings file for the account. Must be UTF-8 encoded as internally it is converted to QString via QString::fromUtf8
settingsNameSize	Size of the settingsName string. If negative (the default), the settingsName size is taken to be strlen(settingsName)

4.2.2.4 ~ApplicationSettings()

```
virtual quentier::ApplicationSettings::~ApplicationSettings ( )  [override], [virtual]
```

Destructor

4.2.3 Member Function Documentation

4.2.3.1 beginGroup() [1/2]

```
void quentier::ApplicationSettings::beginGroup (
            const QString & prefix )
```

Appends prefix to the current group. The call is redirected to QSettings::beginGroup. It is required in this class only to workaround hiding QSettings method due to overloads

Parameters

prefix	String containing the prefix name

4.2.3.2 beginGroup() [2/2]

```
void quentier::ApplicationSettings::beginGroup (
            const char * prefix,
            const int size = -1 )
```

Appends prefix to the current group. Overload of beginGroup accepting const char * and optionally the size of the string

Parameters

prefix	String containing the prefix name. Must be UTF-8 encoded as internally it is converted to QString via QString::fromUtf8
size	Size of the prefix sring. If negative (the default), the prefix size is taken to be stren(prefix).

4.2.3.3 beginReadArray() [1/2]

```
int quentier::ApplicationSettings::beginReadArray (
            const QString & prefix )
```

Adds prefix to the current group and starts reading from an array. The call is redirected to QSettings::beginRead↩ Array. It is required in this class only to workaround hiding QSettings method due to overloads

Parameters

prefix	String containing the prefix name

Returns

The size of the array

4.2.3.4 beginReadArray() [2/2]

```
int quentier::ApplicationSettings::beginReadArray (
            const char * prefix,
            const int size = -1 )
```

Adds prefix to the current group and starts reading from an array. Overload of beginReadArray accepting const char * and optionally the size of the string

Parameters

prefix	String containing the prefix name. Must be UTF-8 encoded as internally it is converted to QString via QString::fromUtf8
size	Size of the prefix sring. If negative (the default), the prefix size is taken to be stren(prefix)

4.2.3.5 beginWriteArray() [1/2]

```
void quentier::ApplicationSettings::beginWriteArray (
            const QString & prefix,
            const int arraySize = -1 )
```

Adds prefix to the current group and starts writing an array of size arraySize. The call is redirected to QSettings↩ ::beginWriteArray. It is required in this class only to workaround hiding QSettings method due to overloads

Parameters

prefix	String containing the prefix name
arraySize	Size of the array to be written. If negative (the default), it is automatically determined based on the indexes of the entries written.

4.2.3.6 beginWriteArray() [2/2]

```
void quentier::ApplicationSettings::beginWriteArray (
            const char * prefix,
            const int arraySize = -1,
            const int prefixSize = -1 )
```

Adds prefix to the current group and starts writing an array of size arraySize. Overload of beginWriteArray accepting const char ∗ and optionally the size of the string

Parameters

prefix	String containing the prefix name. Must be UTF-8 encoded as internally it is converted to QString via QString::fromUtf8
arraySize	Size of the array to be written. If negative (the default), it is automatically determined based on the indexes of the entries written.
prefixSize	Size of the prefix sring. If negative (the default), the prefix size is taken to be stren(prefix)

4.2.3.7 contains() [1/2]

```
bool quentier::ApplicationSettings::contains (
            const QString & key ) const
```

The call is redirected to QSettings::contains. It is required in this class only to workaround hiding QSettings method due to overloads

Parameters

key	The key being checked for presence

Returns

True if there exists a setting called key; false otherwise

4.2.3.8 contains() [2/2]

```
bool quentier::ApplicationSettings::contains (
            const char * key,
            const int size = -1 ) const
```

Overload of contains accepting const char * and optionally the size of the string

Parameters

key	String containing the setting name. Must be UTF-8 encoded as internally it is converted to QString via QString::fromUtf8
size	Size of the key sring. If negative (the default), the key size is taken to be stren(key)

Returns

True if there exists a setting called key; false otherwise

4.2.3.9 remove() [1/2]

```
void quentier::ApplicationSettings::remove (
            const QString & key )
```

Removes the setting key and any sub-settings of key. The call is redirected to QSettings::remove. It is required in this class only to workaround hiding QSettings method due to overloads

Parameters

key	String containing the setting name

4.2.3.10 remove() [2/2]

```
void quentier::ApplicationSettings::remove (
            const char * key,
            const int size = -1 )
```

Removes the setting key and any sub-settings of key. Overload of remove accepting const char * and optionally the size of the string

Parameters

key	String containing the setting name. Must be UTF-8 encoded as internally it is converted to QString via QString::fromUtf8
size	Size of the key sring. If negative (the default), the key size is taken to be stren(key).

4.2.3.11 setValue() [1/2]

```
void quentier::ApplicationSettings::setValue (
            const QString & key,
            const QVariant & value )
```

Sets the value of setting. The call is redirected to QSettings::setValue. It is required in this class only to workaround hiding QSettings method due to overloads

Parameters

key	String containing the setting name
value	Value for setting key

4.2.3.12 setValue() [2/2]

```
void quentier::ApplicationSettings::setValue (
            const char * key,
            const QVariant & value,
            const int keySize = -1 )
```

Sets the value of setting. Overload of setValue accepting const char * and optionally the size of the string

Parameters

key	String containing the setting name. Must be UTF-8 encoded as internally it is converted to QString via QString::fromUtf8
value	Value for setting key
keySize	Size of the key string. If negative (the default), the key size is taken to be strlen(key).

4.2.3.13 value() [1/2]

```
QVariant quentier::ApplicationSettings::value (
            const QString & key,
            const QVariant & defaultValue = {} ) const
```

Fetches the value of setting. The call is redirected to QSettings::value. It is required in this class only to workaround hiding QSettings method due to overloads

Parameters

key	String containing the setting name
defautValue	Default value returned if the setting doesn't exist

Returns

The value for setting key. If the setting doesn't exist, returns defaultValue. If no default value is specified, a default QVariant is returned.

4.2.3.14 value() [2/2]

```
QVariant quentier::ApplicationSettings::value (
            const char * key,
            const QVariant & defaultValue = {},
            const int keySize = -1 ) const
```

Fetches the value of setting. Overload of value accepting const char * and optionally the size of the string

Parameters

key	String containing the setting name. Must be UTF-8 encoded as internally it is converted to QString via QString::fromUtf8
defautValue	Default value returned if the setting doesn't exist
keySize	Size of the key sring. If negative (the default), the key size is taken to be stren(key)

Returns

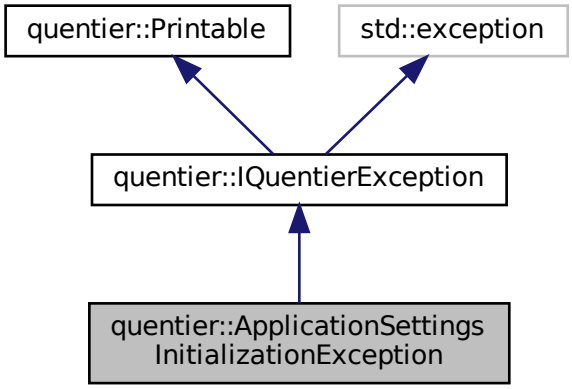
The value for setting key. If the setting doesn't exist, returns defaultValue. If no default value is specified, a default QVariant is returned.

4.3 quentier::ApplicationSettingsInitializationException Class Reference

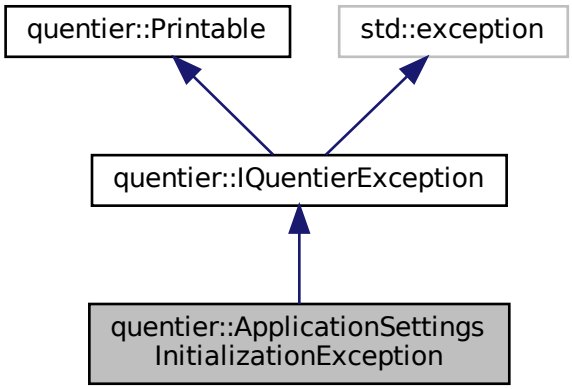
The ApplicationSettingsInitializationException can be thrown from methods of ApplicationSettings class if it's unable to locate the file with persistent settings.

`#include <ApplicationSettingsInitializationException.h>`

Inheritance diagram for quentier::ApplicationSettingsInitializationException:



Collaboration diagram for quentier::ApplicationSettingsInitializationException:



Public Member Functions

- **ApplicationSettingsInitializationException** (const ErrorString &message)

Protected Member Functions

- virtual const QString **exceptionDisplayName** () const override

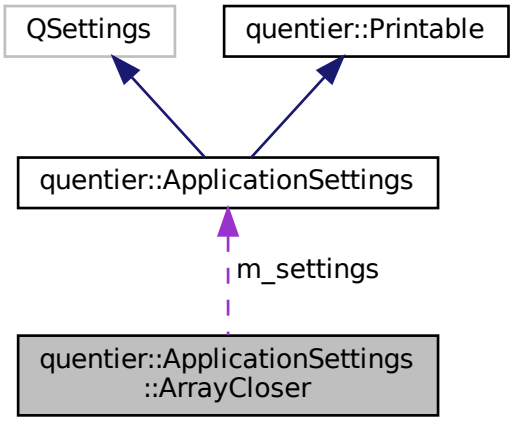
4.3.1 Detailed Description

The ApplicationSettingsInitializationException can be thrown from methods of ApplicationSettings class if it's unable to locate the file with persistent settings.

4.4 quentier::ApplicationSettings::ArrayCloser Struct Reference

`#include <ApplicationSettings.h>`

Collaboration diagram for quentier::ApplicationSettings::ArrayCloser:



Public Member Functions

- **ArrayCloser** (ApplicationSettings &settings)

Public Attributes

- ApplicationSettings & **m_settings**

4.4.1 Detailed Description

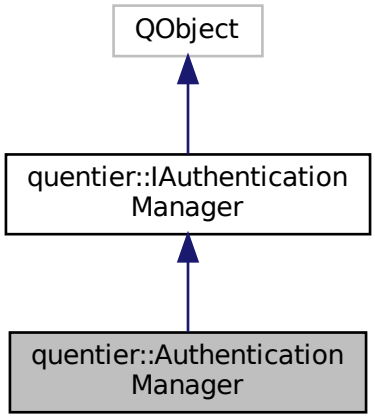
Helper struct for RAII style of ensuring the array once began would be closed even if exception is thrown after beginning the array

4.5 quentier::AuthenticationManager Class Reference

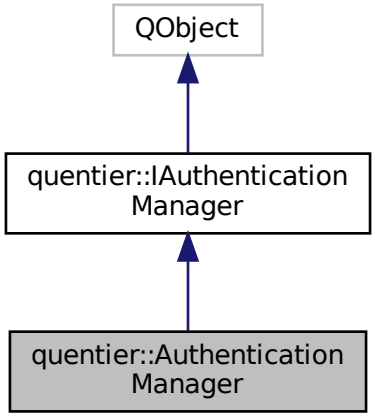
The AuthenticationManager class is libquentier's default implementation of IAuthenticationManager interface; internally uses QEverCloud's OAuth widget.

`#include <AuthenticationManager.h>`

Inheritance diagram for quentier::AuthenticationManager:



Collaboration diagram for quentier::AuthenticationManager:



Public Slots

- virtual void **onAuthenticationRequest** () override

Public Member Functions

- **AuthenticationManager** (const QString &consumerKey, const QString &consumerSecret, const QString &host, QObject *parent=nullptr)

Additional Inherited Members

4.5.1 Detailed Description

The AuthenticationManager class is libquentier's default implementation of IAuthenticationManager interface; internally uses QEverCloud's OAuth widget.

4.6 quentier::ResourceRecognitionIndexItem::BarcodeItem Struct Reference

Public Member Functions

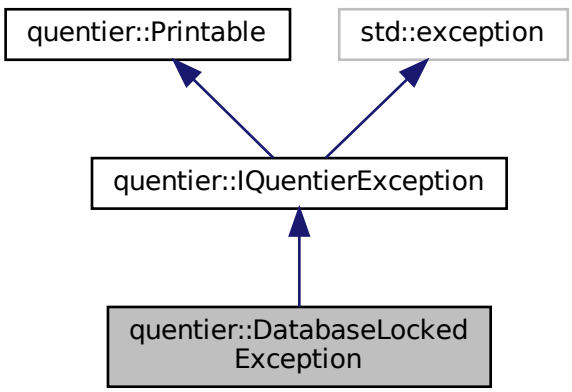
- bool **operator==** (const BarcodeItem &other) const

Public Attributes

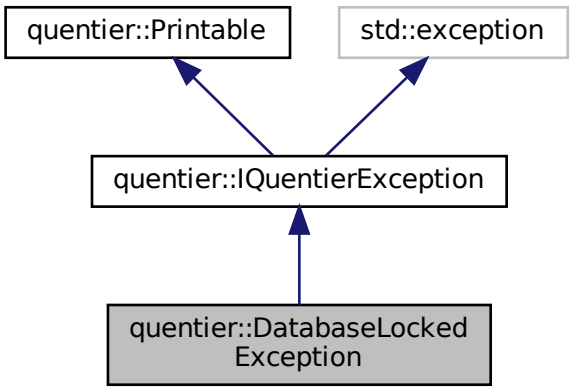
- QString **m_barcode**
- int **m_weight** = -1

4.7 quentier::DatabaseLockedException Class Reference

Inheritance diagram for quentier::DatabaseLockedException:



Collaboration diagram for quentier::DatabaseLockedException:



Public Member Functions

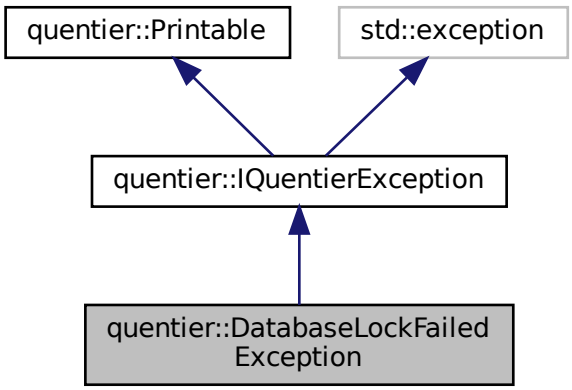
- **DatabaseLockedException** (const ErrorString &message)

Protected Member Functions

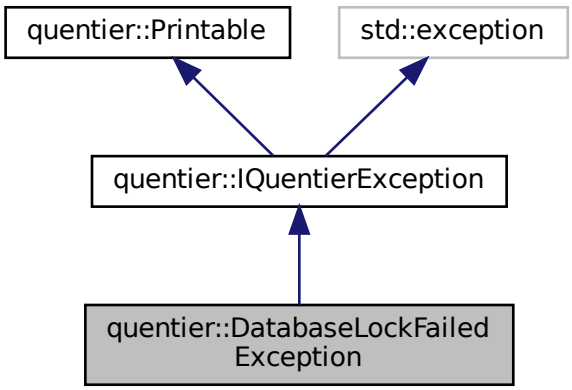
- virtual const QString **exceptionDisplayName** () const override

4.8 quentier::DatabaseLockFailedException Class Reference

Inheritance diagram for quentier::DatabaseLockFailedException:



Collaboration diagram for quentier::DatabaseLockFailedException:



Public Member Functions

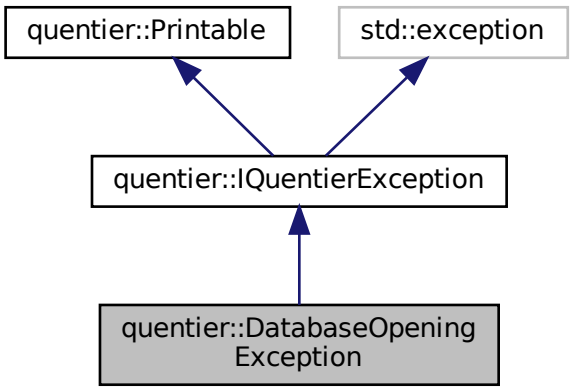
- **DatabaseLockFailedException** (const ErrorString &message)

Protected Member Functions

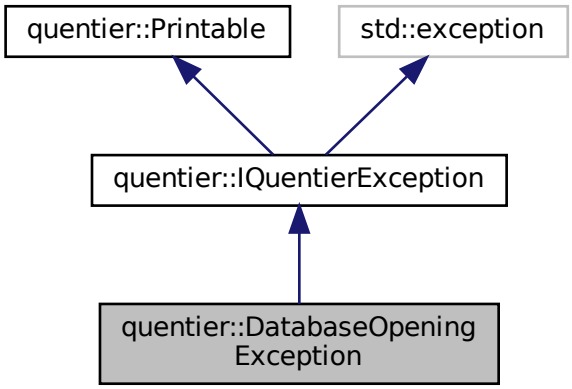
- virtual const QString **exceptionDisplayName** () const override

4.9 quentier::DatabaseOpeningException Class Reference

Inheritance diagram for quentier::DatabaseOpeningException:



Collaboration diagram for quentier::DatabaseOpeningException:



Public Member Functions

- **DatabaseOpeningException** (const ErrorString &message)

Protected Member Functions

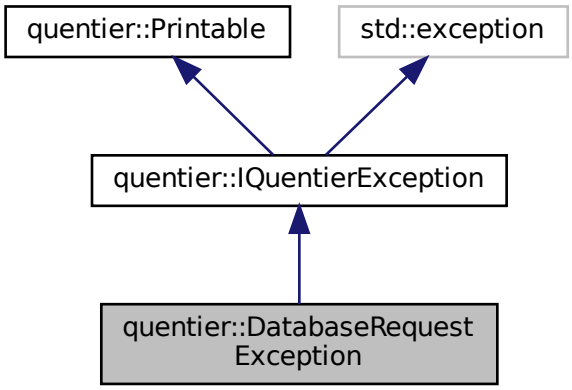
- virtual const QString **exceptionDisplayName** () const override

4.10 quentier::DatabaseRequestException Class Reference

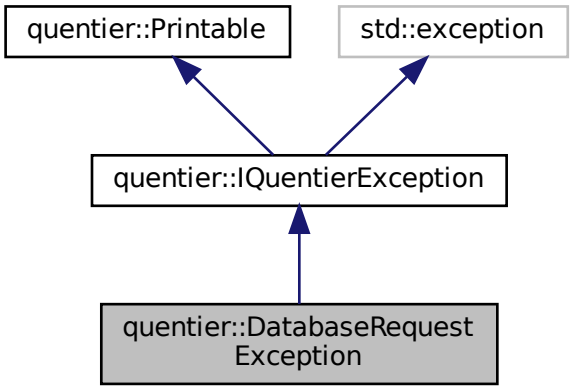
The DatabaseRequestException is thrown when the local storage database encounters some internal error during the attemt to serve a request.

```
#include <DatabaseRequestException.h>
```

Inheritance diagram for quentier::DatabaseRequestException:



Collaboration diagram for quentier::DatabaseRequestException:



Public Member Functions

- **DatabaseRequestException** (const ErrorString &message)

Protected Member Functions

- virtual const QString **exceptionDisplayName** () const override

4.10.1 Detailed Description

The DatabaseRequestException is thrown when the local storage database encounters some internal error during the attemt to serve a request.

4.11 quentier::DateTimePrint Class Reference

The DateTimePrint class simply wraps the enum containing datetime printing options.

`#include <DateTime.h>`

Public Types

- enum Option { IncludeNumericTimestamp = 1 << 1, IncludeMilliseconds = 1 << 2, IncludeTimezone = 1 << 3 }

4.11.1 Detailed Description

The DateTimePrint class simply wraps the enum containing datetime printing options.

4.11.2 Member Enumeration Documentation

4.11.2.1 Option

`enum quentier::DateTimePrint::Option`

Available printing options for datetime

Enumerator

IncludeNumericTimestamp	Include the numeric representation of the timestamp into the printed string
IncludeMilliseconds	Include milliseconds into the printed string
IncludeTimezone	Include timezone into the printed string WARNING: currently this option has no effect on Windows platform, the timezone is not included anyway.

4.12 quentier::DecryptedTextManager Class Reference

Public Member Functions

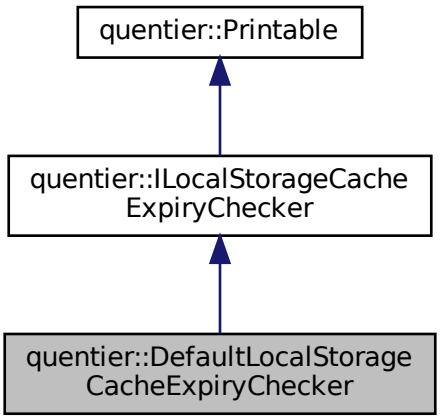
- void **addEntry** (const QString &hash, const QString &decryptedText, const bool rememberForSession, const QString &passphrase, const QString &cipher, const size_t keyLength)

- void **removeEntry** (const QString &hash)
- void **clearNonRememberedForSessionEntries** ()
- bool **findDecryptedTextByEncryptedText** (const QString &encryptedText, QString &decryptedText, bool &rememberForSession) const
- bool **modifyDecryptedText** (const QString &originalEncryptedText, const QString &newDecryptedText, QString &newEncryptedText)

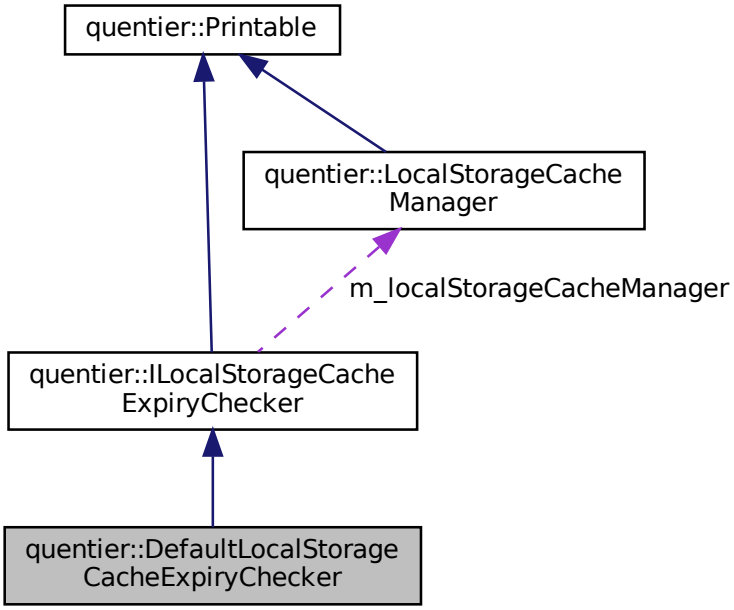
4.13 quentier::DefaultLocalStorageCacheExpiryChecker Class Reference

`#include <DefaultLocalStorageCacheExpiryChecker.h>`

Inheritance diagram for quentier::DefaultLocalStorageCacheExpiryChecker:



Collaboration diagram for quentier::DefaultLocalStorageCacheExpiryChecker:



Public Member Functions

- **DefaultLocalStorageCacheExpiryChecker** (const LocalStorageCacheManager &cacheManager)
- virtual DefaultLocalStorageCacheExpiryChecker * clone () const override
- virtual bool checkNotes () const override
- virtual bool checkResources () const override
- virtual bool checkNotebooks () const override
- virtual bool checkTags () const override
- virtual bool checkLinkedNotebooks () const override
- virtual bool checkSavedSearches () const override
- virtual QTextStream & print (QTextStream &strm) const override

  Print the internal information about the current DefaultLocalStorageCacheExpiryChecker instance to the text stream.

Additional Inherited Members

4.13.1 Detailed Description

brief The DefaultLocalStorageCacheExpiryChecker class is the implementation of ILocalStorageCacheExpiry↩ Checker interface used by LocalStorageCacheManager by default, if no another implementation of ILocalStorage↩ CacheExpiryChecker is set to be used by LocalStorageCacheManager

4.13.2 Member Function Documentation

4.13.2.1 checkLinkedNotebooks()

```
virtual bool quentier::DefaultLocalStorageCacheExpiryChecker::checkLinkedNotebooks ( ) const
[override], [virtual]
```

Returns

False if the current number of cached linked notebooks is higher than a reasonable limit, true otherwise

Implements quentier::ILocalStorageCacheExpiryChecker.

4.13.2.2 checkNotebooks()

```
virtual bool quentier::DefaultLocalStorageCacheExpiryChecker::checkNotebooks ( ) const  [override],
[virtual]
```

Returns

False if the current number of cached notebooks is higher than a reasonable limit, true otherwise

Implements quentier::ILocalStorageCacheExpiryChecker.

4.13.2.3 checkNotes()

```
virtual bool quentier::DefaultLocalStorageCacheExpiryChecker::checkNotes ( ) const  [override],
[virtual]
```

Returns

False if the current number of cached notes is higher than a reasonable limit, true otherwise

Implements quentier::ILocalStorageCacheExpiryChecker.

4.13.2.4 checkResources()

```
virtual bool quentier::DefaultLocalStorageCacheExpiryChecker::checkResources ( ) const  [override],
[virtual]
```

Returns

False if the current number of cached resource is higher than a reasonable limit, true otherwise

Implements quentier::ILocalStorageCacheExpiryChecker.

4.13.2.5 checkSavedSearches()

```
virtual bool quentier::DefaultLocalStorageCacheExpiryChecker::checkSavedSearches ( ) const
[override], [virtual]
```

Returns

False if the current number of cached saved searches is higher than a reasonable limit, true otherwise

Implements quentier::ILocalStorageCacheExpiryChecker.

4.13.2.6 checkTags()

```
virtual bool quentier::DefaultLocalStorageCacheExpiryChecker::checkTags ( ) const  [override],
[virtual]
```

Returns

False if the current number of cached tags is higher than a reasonable limit, true otherwise

Implements quentier::ILocalStorageCacheExpiryChecker.

4.13.2.7 clone()

```
virtual DefaultLocalStorageCacheExpiryChecker* quentier::DefaultLocalStorageCacheExpiry↩
Checker::clone ( ) const  [override], [virtual]
```

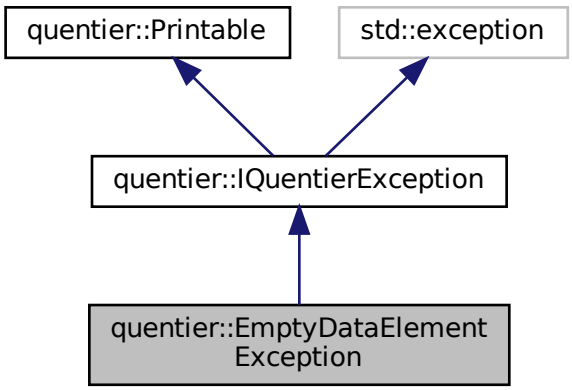
Returns

A pointer to the newly allocated copy of the current DefaultLocalStorageCacheExpiryChecker

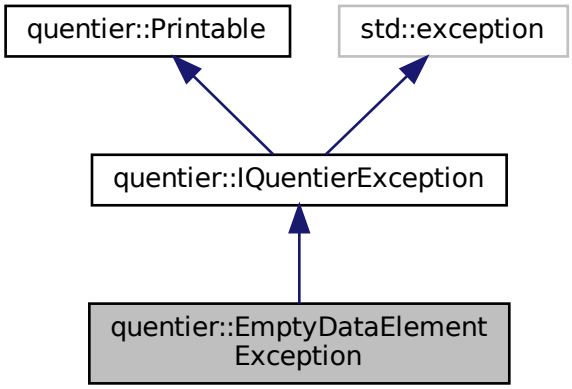
Implements quentier::ILocalStorageCacheExpiryChecker.

4.14 quentier::EmptyDataElementException Class Reference

Inheritance diagram for quentier::EmptyDataElementException:



Collaboration diagram for quentier::EmptyDataElementException:



Public Member Functions

- **EmptyDataElementException** (const ErrorString &message)

Protected Member Functions

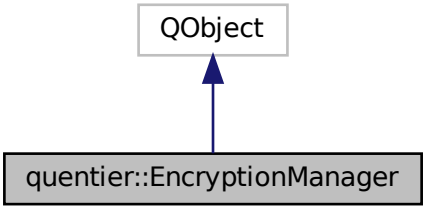
- virtual const QString **exceptionDisplayName** () const override

4.15 quentier::EncryptionManager Class Reference

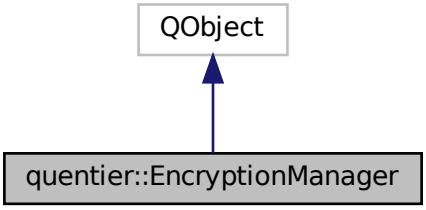
The EncryptionManager class provides both synchronous methods to encrypt or decrypt given text with password, cipher and key length and their signal-slot based potentially asynchronous counterparts.

`#include <EncryptionManager.h>`

Inheritance diagram for quentier::EncryptionManager:



Collaboration diagram for quentier::EncryptionManager:



Public Slots

- void **onDecryptTextRequest** (QString encryptedText, QString passphrase, QString cipher, size_t keyLength, QUuid requestId)
- void **onEncryptTextRequest** (QString textToEncrypt, QString passphrase, QString cipher, size_t keyLength, QUuid requestId)

Signals

- void **decryptedText** (QString text, bool success, ErrorString errorDescription, QUuid requestId)
- void **encryptedText** (QString encryptedText, bool success, ErrorString errorDescription, QUuid requestId)

Public Member Functions

- **EncryptionManager** (QObject *parent=nullptr)
- bool **decrypt** (const QString &encryptedText, const QString &passphrase, const QString &cipher, const size_t keyLength, QString &decryptedText, ErrorString &errorDescription)
- bool **encrypt** (const QString &textToEncrypt, const QString &passphrase, QString &cipher, size_t &keyLength, QString &encryptedText, ErrorString &errorDescription)

4.15.1 Detailed Description

The EncryptionManager class provides both synchronous methods to encrypt or decrypt given text with password, cipher and key length and their signal-slot based potentially asynchronous counterparts.

4.16 quentier::ENMLConverter Class Reference

The ENMLConverter class encapsulates a set of methods and helper data structures for performing the conversions between ENML and other note content formats, namely HTML.

```
#include <ENMLConverter.h>
```

Classes

- struct NoteContentToHtmlExtraData
- class SkipHtmlElementRule

  The SkipHtmlElementRule class describes the set of rules for HTML -> ENML conversion about the HTML elements that should not be actually converted to ENML due to their nature of being "helper" elements for the display or functioning of something within the note editor's page. The HTML -> ENML conversion would ignore tags and attributes forbidden by ENML even without these rules conditionally preserving or skipping the contents and nested elements of skipped elements.

Public Types

- enum EnexExportTags { **Yes** = 0, **No** }

  The EnexExportTags enum allows to specify whether export of note(s) to ENEX should include the names of note's tags.

Public Member Functions

- bool **htmlToNoteContent** (const QString &html, QString &noteContent, DecryptedTextManager &decryptedTextManager, ErrorString &errorDescription, const QVector< SkipHtmlElementRule > &skipRules={}) const
- bool cleanupExternalHtml (const QString &inputHtml, QString &cleanedUpHtml, ErrorString &errorDescription) const

  cleanupExternalHtml method cleans up a piece of HTML coming from some external source: the cleanup includes the removal (or replacement with equivalents/alternatives) of any tags and attributes not supported by the ENML representation of note page's HTML
- bool htmlToQTextDocument (const QString &html, QTextDocument &doc, ErrorString &errorDescription, const QVector< SkipHtmlElementRule > &skipRules={}) const

- bool **noteContentToHtml** (const QString &noteContent, QString &html, ErrorString &errorDescription, DecryptedTextManager &decryptedTextManager, NoteContentToHtmlExtraData &extraData) const
- bool **validateEnml** (const QString &enml, ErrorString &errorDescription) const
- bool **validateAndFixupEnml** (QString &enml, ErrorString &errorDescription) const
- bool exportNotesToEnex (const QVector< Note > &notes, const QHash< QString, QString > &tagNamesByTagLocalUids, const EnexExportTags exportTagsOption, QString &enex, ErrorString &errorDescription, const QString &version={}) const

  exportNotesToEnex exports either a single note or a set of notes into ENEX format
- bool importEnex (const QString &enex, QVector< Note > &notes, QHash< QString, QStringList > &tagNamesByNoteLocalUid, ErrorString &errorDescription) const

  importEnex reads the content of input ENEX file and converts it into a set of notes and tag names.

Static Public Member Functions

- static bool **noteContentToPlainText** (const QString &noteContent, QString &plainText, ErrorString &errorMessage)
- static bool **noteContentToListOfWords** (const QString &noteContent, QStringList &listOfWords, ErrorString &errorMessage, QString *plainText=nullptr)
- static QStringList **plainTextToListOfWords** (const QString &plainText)
- static QString **toDoCheckboxHtml** (const bool checked, const quint64 idNumber)
- static QString **encryptedTextHtml** (const QString &encryptedText, const QString &hint, const QString &cipher, const size_t keyLength, const quint64 enCryptIndex)
- static QString **decryptedTextHtml** (const QString &decryptedText, const QString &encryptedText, const QString &hint, const QString &cipher, const size_t keyLength, const quint64 enDecryptedIndex)
- static QString **resourceHtml** (const Resource &resource, ErrorString &errorDescription)
- static void **escapeString** (QString &string, const bool simplify=true)

4.16.1 Detailed Description

The ENMLConverter class encapsulates a set of methods and helper data structures for performing the conversions between ENML and other note content formats, namely HTML.

4.16.2 Member Function Documentation

4.16.2.1 cleanupExternalHtml()

```
bool quentier::ENMLConverter::cleanupExternalHtml (
            const QString & inputHtml,
            QString & cleanedUpHtml,
            ErrorString & errorDescription ) const
```

cleanupExternalHtml method cleans up a piece of HTML coming from some external source: the cleanup includes the removal (or replacement with equivalents/alternatives) of any tags and attributes not supported by the ENML representation of note page's HTML

Parameters

inputHtml	- the input HTML to be cleaned up
cleanedUpHtml	- the result of the method's work
errorDescription	- the textual description of the error if conversion of input HTML into QTextDocument has failed

Returns

true in case of successful conversion, false otherwise

4.16.2.2 exportNotesToEnex()

```
bool quentier::ENMLConverter::exportNotesToEnex (
            const QVector< Note > & notes,
            const QHash< QString, QString > & tagNamesByTagLocalUids,
            const EnexExportTags exportTagsOption,
            QString & enex,
            ErrorString & errorDescription,
            const QString & version = {} ) const
```

exportNotesToEnex exports either a single note or a set of notes into ENEX format

Parameters

notes	The notes to be exported into the enex format. The connection of particular notes to tags is expected to follow from note's tag local uids. In other words, if some note has no tag local uids, its corresponding fragment of ENEX won't contain tag names associated with the note
tagNamesByTagLocalUids	Tag names for all tag local uids across all passed in notes. The lack of any tag name for any tag local uid is considered an error and the overall export attempt fails
exportTagsOption	Whether the export to ENEX should include the names of notes' tags
enex	The output of the method
errorDescription	The textual description of the error, if any
version	Optional "version" tag for the ENEX. If not set, the corresponding ENEX tag is set to empty value

Returns

True if the export completed successfully, false otherwise

4.16.2.3 htmlToQTextDocument()

```
bool quentier::ENMLConverter::htmlToQTextDocument (
            const QString & html,
            QTextDocument & doc,
            ErrorString & errorDescription,
            const QVector< SkipHtmlElementRule > & skipRules = {} ) const
```

Converts the passed in HTML into its simplified form acceptable by QTextDocument (see http://doc.qt.↩io/qt-5/richtext-html-subset.html for the list of elements supported by QTextDocument)

Parameters

html	- the input HTML which needs to be converted to QTextDocument
doc	- QTextDocument filled with the result of the method's work
errorDescription	- the textual description of the error if conversion of input HTML into QTextDocument has failed
skipRules	- rules for skipping the particular elements

Returns

true in case of successful conversion, false otherwise

4.16.2.4 importEnex()

```
bool quentier::ENMLConverter::importEnex (
            const QString & enex,
            QVector< Note > & notes,
            QHash< QString, QStringList > & tagNamesByNoteLocalUid,
            ErrorString & errorDescription ) const
```

importEnex reads the content of input ENEX file and converts it into a set of notes and tag names.

Parameters

enex	The input ENEX file contents
notes	Notes read from the ENEX
tagNamesByNoteLocalUid	Tag names per each read note; it is the responsibility of the method caller to find the actual tags corresponding to these names and set the tag local uids and/or guids to the note
errorDescription	The textual descrition of the error if the ENEX file could not be read and converted into a set of notes and tag names for them

Returns

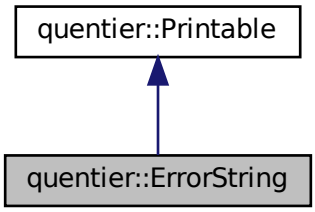
True of the ENEX file was read and converted into a set of notes and tag names successfully, false otherwise

4.17 quentier::ErrorString Class Reference

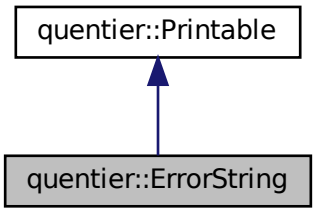
The ErrorString class encapsulates two (or more) strings which are meant to contain translatable (base) and non-translatable (details) parts of the error description.

```
#include <ErrorString.h>
```

Inheritance diagram for quentier::ErrorString:



Collaboration diagram for quentier::ErrorString:



Public Member Functions

- **ErrorString** (const char ∗error=nullptr)
- **ErrorString** (const QString &error)
- **ErrorString** (const ErrorString &other)
- ErrorString & **operator=** (const ErrorString &other)
- const QString & **base** () const
- QString & **base** ()
- const QStringList & **additionalBases** () const
- QStringList & **additionalBases** ()
- const QString & **details** () const
- QString & **details** ()
- void **setBase** (const QString &error)
- void **setBase** (const char ∗error)
- void **appendBase** (const QString &error)
- void **appendBase** (const QStringList &errors)
- void **appendBase** (const char ∗error)
- void **setDetails** (const QString &error)
- void **setDetails** (const char ∗error)
- bool **isEmpty** () const
- void **clear** ()
- QString **localizedString** () const
- QString **nonLocalizedString** () const
- virtual QTextStream & **print** (QTextStream &strm) const override

Additional Inherited Members

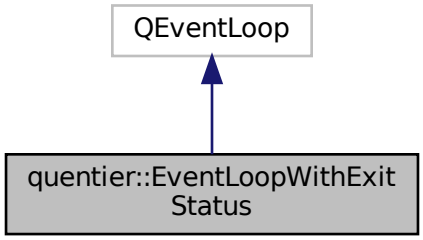
4.17.1 Detailed Description

The ErrorString class encapsulates two (or more) strings which are meant to contain translatable (base) and non-translatable (details) parts of the error description.

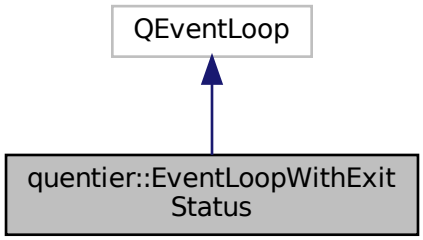
1. base() methods return const and non-const links to the primary translatable string
2. details() methods ruturn const and non-const links to non-translatable string (coming from some third party library etc)
3. additionalBases() methods return const and non-const links to additional translatable strings; one translatable string is not always enough because the error message might be composed from different parts

4.18 quentier::EventLoopWithExitStatus Class Reference

Inheritance diagram for quentier::EventLoopWithExitStatus:



Collaboration diagram for quentier::EventLoopWithExitStatus:



Public Types

- enum **ExitStatus** { **Success**, **Failure**, **Timeout** }

Public Slots

- void **exitAsSuccess** ()
- void **exitAsFailure** ()
- void **exitAsFailureWithError** (QString errorDescription)
- void **exitAsFailureWithErrorString** (ErrorString errorDescription)
- void **exitAsTimeout** ()

Public Member Functions

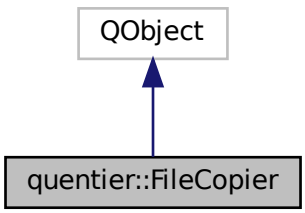
- **EventLoopWithExitStatus** (QObject *parent=nullptr)
- ExitStatus **exitStatus** () const
- const ErrorString & **errorDescription** () const

Friends

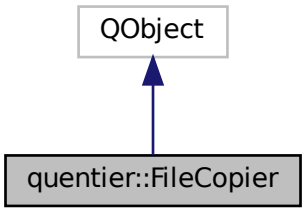
- QDebug & **operator<<** (QDebug &dbg, const ExitStatus status)
- QTextStream & **operator<<** (QTextStream &strm, const ExitStatus status)

4.19 quentier::FileCopier Class Reference

Inheritance diagram for quentier::FileCopier:



Collaboration diagram for quentier::FileCopier:



Public Types

- enum **State** { **Idle** = 0, **Copying**, **Cancelling** }

Public Slots

- void **copyFile** (QString sourcePath, QString destPath)
- void **cancel** ()

Signals

- void **progressUpdate** (double progress)
- void **finished** (QString sourcePath, QString destPath)
- void **cancelled** (QString sourcePath, QString destPath)
- void **notifyError** (ErrorString error)

Public Member Functions

- **FileCopier** (QObject *parent=nullptr)
- State **state** () const
- QString **sourceFilePath** () const
- QString **destinationFilePath** () const
- double **currentProgress** () const

Friends

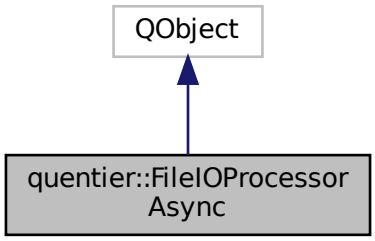
- QDebug & **operator<<** (QDebug &dbg, const State state)
- QTextStream & **operator<<** (QTextStream &strm, const State state)

4.20 quentier::FileIOProcessorAsync Class Reference

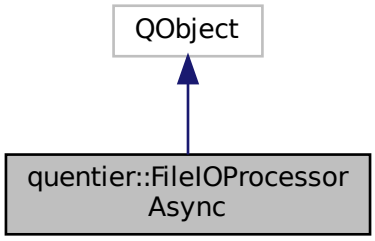
The FileIOProcessorAsync class is a wrapper under simple file IO operations, it is meant to be used for simple asynchronous IO.

`#include <FileIOProcessorAsync.h>`

Inheritance diagram for quentier::FileIOProcessorAsync:



Collaboration diagram for quentier::FileIOProcessorAsync:



Public Slots

- void onWriteFileRequest (QString absoluteFilePath, QByteArray data, QUuid requestId, bool append)

  onWriteFileRequest slot processes file write requests with given request ids
- void onReadFileRequest (QString absoluteFilePath, QUuid requestId)

  onReadFileRequest slot processes file read requests with given request ids

Signals

- void readyForIO ()

  readyForIO signal is emitted when the queue for file IO is empty for some time (30 seconds by default, can also be configured via setIdleTimePeriod method) after the last IO event to signal listeners that they can perform some IO via the FileIOProcessorAsync
- void writeFileRequestProcessed (bool success, ErrorString errorDescription, QUuid requestId)

  writeFileRequestProcessed signal is emitted when the file write request with given id is finished
- void readFileRequestProcessed (bool success, ErrorString errorDescription, QByteArray data, QUuid requestId)

  readFileRequestProcessed signal is emitted when the file read request with given id is finished

Public Member Functions

- **FileIOProcessorAsync** (QObject *parent=nullptr)
- void setIdleTimePeriod (qint32 seconds)

  setIdleTimePeriod sets time period defining the idle state of FileIOProcessorAsync: once the time measured since the last IO operation is over the specified number of seconds, the class emits readyForIO signal to any interested listeners of this event. If this method is not called ever, the default idle time period would be 30 seconds.

4.20.1 Detailed Description

The FileIOProcessorAsync class is a wrapper under simple file IO operations, it is meant to be used for simple asynchronous IO.

4.20.2 Member Function Documentation

4.20.2.1 onReadFileRequest

```
void quentier::FileIOProcessorAsync::onReadFileRequest (
            QString absoluteFilePath,
            QUuid requestId ) [slot]
```

onReadFileRequest slot processes file read requests with given request ids

Parameters

absoluteFilePath	Absolute file path to be read
requestId	Unique identifier of the file read request

4.20.2.2 onWriteFileRequest

```
void quentier::FileIOProcessorAsync::onWriteFileRequest (
            QString absoluteFilePath,
```

```
            QByteArray data,
            QUuid requestId,
            bool append )  [slot]
```

onWriteFileRequest slot processes file write requests with given request ids

Parameters

absoluteFilePath	Absolute file path to be written
data	Data to be written to the file
requestId	Unique identifier of the file write request
append	If true, the data would be appended to file, otherwise the entire file would be erased before with the data is written

4.20.2.3 readFileRequestProcessed

```
void quentier::FileIOProcessorAsync::readFileRequestProcessed (
            bool success,
            ErrorString errorDescription,
            QByteArray data,
            QUuid requestId )  [signal]
```

readFileRequestProcessed signal is emitted when the file read request with given id is finished

Parameters

success	True if read operation was successful, false otherwise
errorDescription	Textual description of the error
data	Data read from file
requestId	Unique identifier of the file read request

4.20.2.4 setIdleTimePeriod()

```
void quentier::FileIOProcessorAsync::setIdleTimePeriod (
            qint32 seconds )
```

setIdleTimePeriod sets time period defining the idle state of FileIOProcessorAsync: once the time measured since the last IO operation is over the specified number of seconds, the class emits readyForIO signal to any interested listeners of this event. If this method is not called ever, the default idle time period would be 30 seconds.

Parameters

seconds	Number of seconds for idle time period

4.20.2.5 writeFileRequestProcessed

```
void quentier::FileIOProcessorAsync::writeFileRequestProcessed (
            bool success,
            ErrorString errorDescription,
            QUuid requestId )  [signal]
```

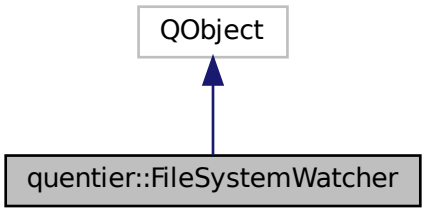
writeFileRequestProcessed signal is emitted when the file write request with given id is finished

Parameters

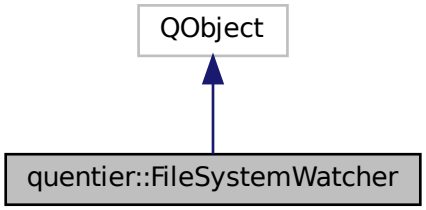
success	True if write operation was successful, false otherwise
errorDescription	Textual description of the error
requestId	Unique identifier of the file write request

4.21 quentier::FileSystemWatcher Class Reference

Inheritance diagram for quentier::FileSystemWatcher:



Collaboration diagram for quentier::FileSystemWatcher:



Signals

- void **directoryChanged** (const QString &path)
- void **directoryRemoved** (const QString &path)
- void **fileChanged** (const QString &path)
- void **fileRemoved** (const QString &path)

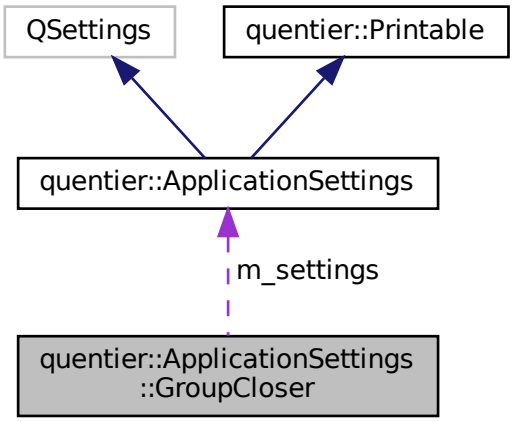
Public Member Functions

- **FileSystemWatcher** (const int removalTimeoutMSec=FILE_SYSTEM_WATCHER_DEFAULT_REMOVA↩ L_TIMEOUT_MSEC, QObject *parent=nullptr)
- **FileSystemWatcher** (const QStringList &paths, const int removalTimeoutMSec=FILE_SYSTEM_WATCH↩ ER_DEFAULT_REMOVAL_TIMEOUT_MSEC, QObject *parent=nullptr)
- void **addPath** (const QString &path)
- void **addPaths** (const QStringList &paths)
- QStringList **directories** () const
- QStringList **files** () const
- void **removePath** (const QString &path)
- void **removePaths** (const QStringList &paths)

4.22 quentier::ApplicationSettings::GroupCloser Struct Reference

`#include <ApplicationSettings.h>`

Collaboration diagram for quentier::ApplicationSettings::GroupCloser:



Public Member Functions

- **GroupCloser** (ApplicationSettings &settings)

Public Attributes

- ApplicationSettings & **m_settings**

4.22.1 Detailed Description

Helper struct for RAII style of ensuring the group once opened would be closed even if exception is thrown after beginning the group

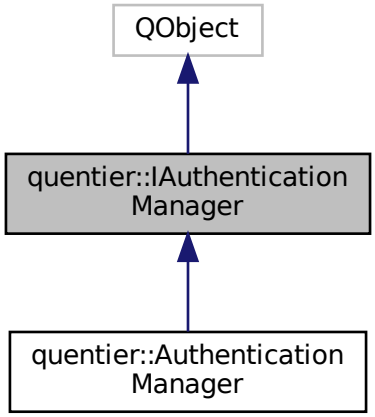
4.23 quentier::HTMLCleaner Class Reference

Public Member Functions

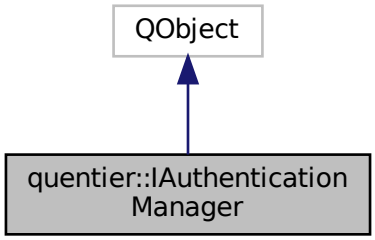
- bool **htmlToXml** (const QString &html, QString &output, QString &errorDescription)
- bool **htmlToXhtml** (const QString &html, QString &output, QString &errorDescription)
- bool **cleanupHtml** (QString &html, QString &errorDescription)

4.24 quentier::IAuthenticationManager Class Reference

Inheritance diagram for quentier::IAuthenticationManager:



Collaboration diagram for quentier::IAuthenticationManager:



Public Slots

- virtual void **onAuthenticationRequest** ()=0

Signals

- void **sendAuthenticationResult** (bool success, qevercloud::UserID userId, QString authToken, qevercloud::Timestamp authTokenExpirationTime, QString shardId, QString noteStoreUrl, QString webApi↩ UrlPrefix, QList< QNetworkCookie > userStoreCookies, ErrorString errorDescription)

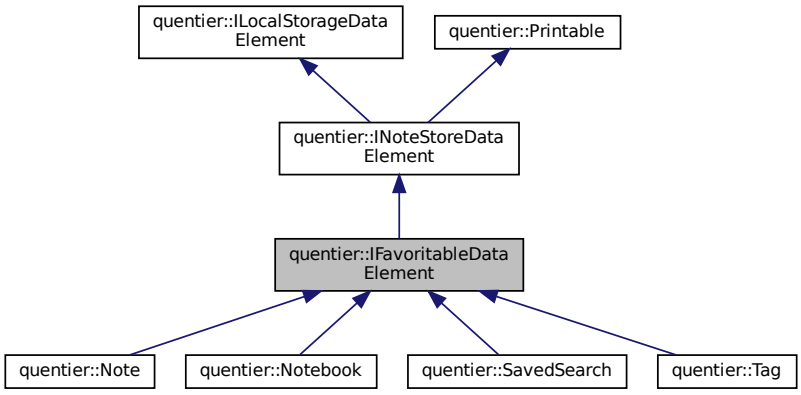
Protected Member Functions

- **IAuthenticationManager** (QObject *parent=nullptr)

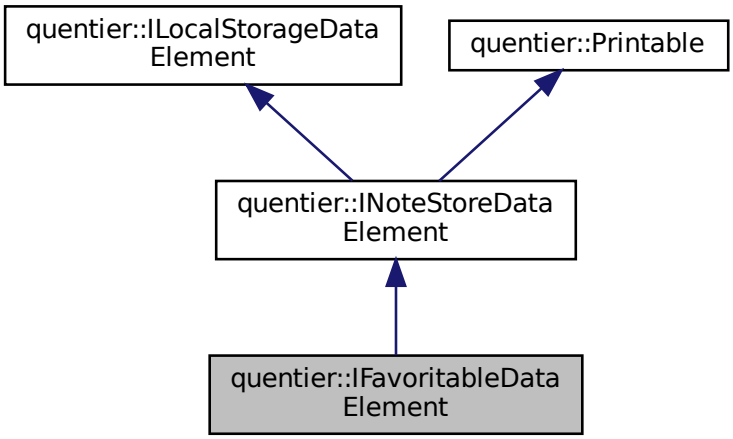
4.25 quentier::IFavoritableDataElement Class Reference

`#include <IFavoritableDataElement.h>`

Inheritance diagram for quentier::IFavoritableDataElement:



Collaboration diagram for quentier::IFavoritableDataElement:



Public Member Functions

- virtual bool **isFavorited** () const =0
- virtual void **setFavorited** (const bool favorited)=0

Additional Inherited Members

4.25.1 Detailed Description

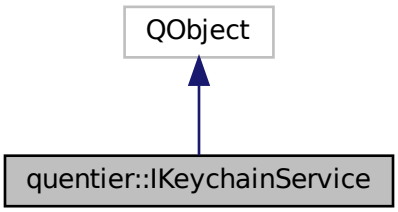
Class adding one more attribute to each note store data element subclassing it: "favorited" flag. Favorited data elements are expected to be somehow arranged for quick access in the client application's UI.

4.26 quentier::IKeychainService Class Reference

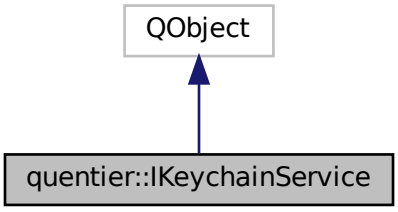
The IKeychainService interface provides methods intended to start potentially asynchronous interaction with the keychain and signals intended to notify listeners about the completion of asynchronous interactions.

```
#include <IKeychainService.h>
```

Inheritance diagram for quentier::IKeychainService:



Collaboration diagram for quentier::IKeychainService:



Public Types

- enum ErrorCode {
  ErrorCode::NoError, ErrorCode::EntryNotFound, ErrorCode::CouldNotDeleteEntry, ErrorCode::Access↩
  DeniedByUser,
  ErrorCode::AccessDenied, ErrorCode::NoBackendAvailable, ErrorCode::NotImplemented, ErrorCode::↩
  OtherError }

Signals

- void writePasswordJobFinished (QUuid requestId, ErrorCode errorCode, ErrorString errorDescription)
- void readPasswordJobFinished (QUuid requestId, ErrorCode errorCode, ErrorString errorDescription, Q↩
  String password)
- void deletePasswordJobFinished (QUuid requestId, ErrorCode errorCode, ErrorString errorDescription)

Public Member Functions

- virtual QUuid startWritePasswordJob (const QString &service, const QString &key, const QString &password)=0
- virtual QUuid startReadPasswordJob (const QString &service, const QString &key)=0
- virtual QUuid startDeletePasswordJob (const QString &service, const QString &key)=0

Protected Member Functions

- **IKeychainService** (QObject *parent=nullptr)

Friends

- QTextStream & **operator<<** (QTextStream &strm, const ErrorCode errorCode)
- QDebug & **operator<<** (QDebug &dbg, const ErrorCode errorCode)

4.26.1 Detailed Description

The IKeychainService interface provides methods intended to start potentially asynchronous interaction with the keychain and signals intended to notify listeners about the completion of asynchronous interactions.

4.26.2 Member Enumeration Documentation

4.26.2.1 ErrorCode

```
enum quentier::IKeychainService::ErrorCode  [strong]
```

Error codes for results of operations with the keychain service

Enumerator

NoError	No error occurred, operation was successful
EntryNotFound	For the given key no data was found
CouldNotDeleteEntry	Could not delete existing secret data
AccessDeniedByUser	User denied access to keychain
AccessDenied	Access denied for some reason
NoBackendAvailable	No platform-specific keychain service available
NotImplemented	Not implemented on platform
OtherError	Something else went wrong, the error description specifies what

4.26.3 Member Function Documentation

4.26.3.1 deletePasswordJobFinished

```
void quentier::IKeychainService::deletePasswordJobFinished (
            QUuid requestId,
```

```
ErrorCode errorCode,
ErrorString errorDescription )  [signal]
```

deletePasswordJobFinished signal should be emitted in response to the call of startDeletePasswordJob method

Parameters

requestId	Request id returned from startDeletePasswordJob method
errorCode	Error code determining whether the operation was successful or some error has occurred
errorDescription	Textual description of error in case of unsuccessful execution

4.26.3.2 readPasswordJobFinished

```
void quentier::IKeychainService::readPasswordJobFinished (
            QUuid requestId,
            ErrorCode errorCode,
            ErrorString errorDescription,
            QString password )  [signal]
```

readPasswordJobFinished signal should be emitted in response to the call of startReadPasswordJob method

Parameters

requestId	Request id returned from startReadPasswordJob method
errorCode	Error code determining whether the operation was successful or some error has occurred
errorDescription	Textual description of error in case of unsuccessful execution
password	Password read from the keychain

4.26.3.3 startDeletePasswordJob()

```
virtual QUuid quentier::IKeychainService::startDeletePasswordJob (
            const QString & service,
            const QString & key )  [pure virtual]
```

startDeletePasswordJob slot should start the potentially asynchronous process of deleting the password from the keychain. When ready, this slot is expected to emit deletePasswordJobFinished signal.

Parameters

service	Name of service within the keychain
key	Key under which the password is stored

Returns

Unique identifier assigned to this delete password request

4.26.3.4 startReadPasswordJob()

```
virtual QUuid quentier::IKeychainService::startReadPasswordJob (
            const QString & service,
            const QString & key )  [pure virtual]
```

startReadPasswordJob slot should start the potentially asynchronous process of reading the password from the keychain. When ready, this slot is expected to emit readPasswordJobFinished signal.

Parameters

service	Name of service within the keychain
key	Key under which the password is stored

Returns

Unique identifier assigned to this read password request

4.26.3.5 startWritePasswordJob()

```
virtual QUuid quentier::IKeychainService::startWritePasswordJob (
            const QString & service,
            const QString & key,
            const QString & password )  [pure virtual]
```

startWritePasswordJob slot should start the potentially asynchronous process of storing the password in the keychain. When ready, this slot is expected to emit writePasswordJobFinished signal.

Parameters

service	Name of service within the keychain
key	Key to store the password under
password	Password to store in the keychain

Returns

Unique identifier assigned to this write password request

4.26.3.6 writePasswordJobFinished

```
void quentier::IKeychainService::writePasswordJobFinished (
            QUuid requestId,
            ErrorCode errorCode,
            ErrorString errorDescription )  [signal]
```

writePasswordJobFinished signal should be emitted in response to the call of startWritePasswordJob method

Parameters

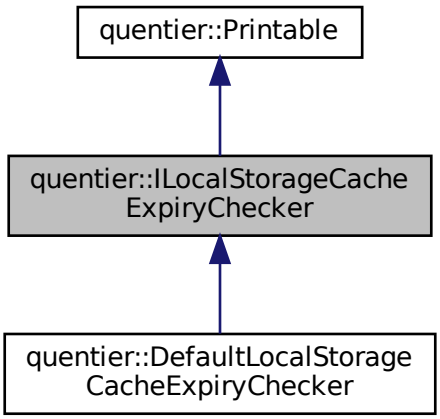
requestId	Request id returned from startWritePasswordJob method
errorCode	Error code determining whether the operation was successful or some error has occurred
errorDescription	Textual description of error in case of unsuccessful execution

4.27 quentier::ILocalStorageCacheExpiryChecker Class Reference

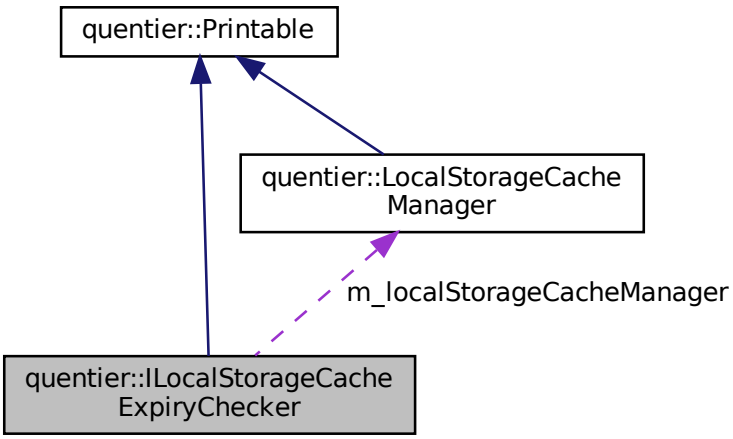
The ILocalStorageCacheExpiryChecker class represents the interface for cache expiry checker used by LocalStorageCacheManager to see whether particular caches (of notes, notebooks, tags, linked notebooks and/or saved searches) need to be shrunk.

```
#include <ILocalStorageCacheExpiryChecker.h>
```

Inheritance diagram for quentier::ILocalStorageCacheExpiryChecker:



Collaboration diagram for quentier::ILocalStorageCacheExpiryChecker:



Public Member Functions

- virtual ILocalStorageCacheExpiryChecker * clone () const =0
- virtual bool checkNotes () const =0
- virtual bool checkResources () const =0
- virtual bool checkNotebooks () const =0
- virtual bool checkTags () const =0
- virtual bool checkLinkedNotebooks () const =0
- virtual bool checkSavedSearches () const =0
- virtual QTextStream & print (QTextStream &strm) const =0

  Print the internal information about ILocalStorageCacheExpiryChecker implementation instance to the text stream.

Protected Member Functions

- **ILocalStorageCacheExpiryChecker** (const LocalStorageCacheManager &cacheManager)

Protected Attributes

- const LocalStorageCacheManager & **m_localStorageCacheManager**

4.27.1 Detailed Description

The ILocalStorageCacheExpiryChecker class represents the interface for cache expiry checker used by LocalStorageCacheManager to see whether particular caches (of notes, notebooks, tags, linked notebooks and/or saved searches) need to be shrunk.

4.27.2 Member Function Documentation

4.27.2.1 checkLinkedNotebooks()

```
virtual bool quentier::ILocalStorageCacheExpiryChecker::checkLinkedNotebooks ( ) const  [pure
virtual]
```

Returns

False if the cache of linked notebooks needs to be shrunk (due to its size or whatever other reason), true otherwise

Implemented in quentier::DefaultLocalStorageCacheExpiryChecker.

4.27.2.2 checkNotebooks()

```
virtual bool quentier::ILocalStorageCacheExpiryChecker::checkNotebooks ( ) const  [pure virtual]
```

Returns

False if the cache of notebooks needs to be shrunk (due to its size or whatever other reason), true otherwise

Implemented in quentier::DefaultLocalStorageCacheExpiryChecker.

4.27.2.3 checkNotes()

```
virtual bool quentier::ILocalStorageCacheExpiryChecker::checkNotes ( ) const  [pure virtual]
```

Returns

False if the cache of notes needs to be shrunk (due to its size or whatever other reason), true otherwise

Implemented in quentier::DefaultLocalStorageCacheExpiryChecker.

4.27.2.4 checkResources()

```
virtual bool quentier::ILocalStorageCacheExpiryChecker::checkResources ( ) const  [pure virtual]
```

Returns

False if the cache of resources needs to be shrunk (due to its size or whatever other reason), true otherwise

Implemented in quentier::DefaultLocalStorageCacheExpiryChecker.

4.27.2.5 checkSavedSearches()

```
virtual bool quentier::ILocalStorageCacheExpiryChecker::checkSavedSearches ( ) const  [pure
virtual]
```

Returns

False if the cache of saved searches needs to be shrunk (due to its size or whatever other reason), true otherwise

Implemented in quentier::DefaultLocalStorageCacheExpiryChecker.

4.27.2.6 checkTags()

```
virtual bool quentier::ILocalStorageCacheExpiryChecker::checkTags ( ) const  [pure virtual]
```

Returns

False if the cache of tags needs to be shrunk (due to its size or whatever other reason), true otherwise

Implemented in quentier::DefaultLocalStorageCacheExpiryChecker.

4.27.2.7 clone()

```
virtual ILocalStorageCacheExpiryChecker* quentier::ILocalStorageCacheExpiryChecker::clone ( )
const  [pure virtual]
```

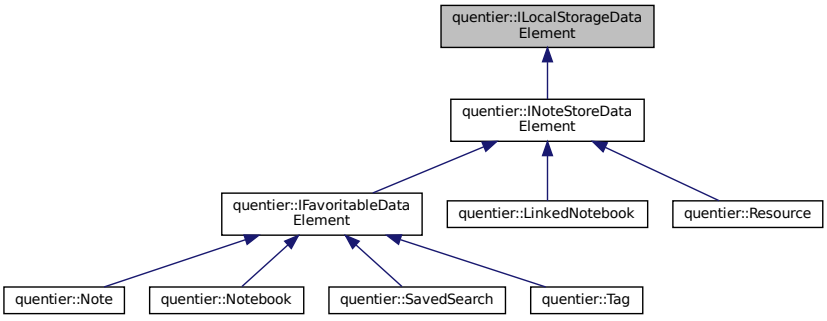
Returns

A pointer to the newly allocated copy of a particular ILocalStorageCacheExpiryChecker implementation

Implemented in quentier::DefaultLocalStorageCacheExpiryChecker.

4.28 quentier::ILocalStorageDataElement Class Reference

Inheritance diagram for quentier::ILocalStorageDataElement:



Public Member Functions

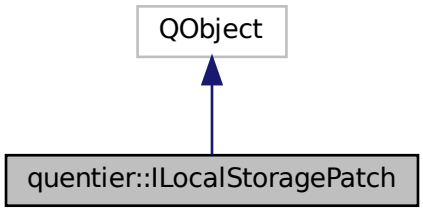
- virtual const QString **localUid** () const =0
- virtual void **setLocalUid** (const QString &guid)=0
- virtual void **unsetLocalUid** ()=0

4.29 quentier::ILocalStoragePatch Class Reference

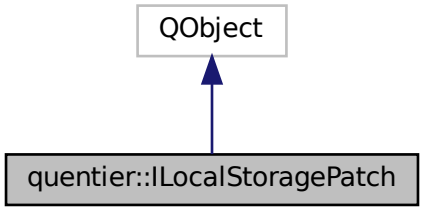
The ILocalStoragePatch class represents the interface for patches of local storage. Each such patch somehow changes the layout of local storage persistence so that only compliant & corresponding versions of libquentier can be used to work with it.

`#include <ILocalStoragePatch.h>`

Inheritance diagram for quentier::ILocalStoragePatch:



Collaboration diagram for quentier::ILocalStoragePatch:



Signals

- void progress (double progress)
- void backupProgress (double progress)
- void restoreBackupProgress (double progress)

Public Member Functions

- virtual int fromVersion () const =0
- virtual int toVersion () const =0
- virtual QString patchShortDescription () const =0
- virtual QString patchLongDescription () const =0
- virtual bool backupLocalStorage (ErrorString &errorDescription)=0
- virtual bool restoreLocalStorageFromBackup (ErrorString &errorDescription)=0
- virtual bool removeLocalStorageBackup (ErrorString &errorDescription)=0
- virtual bool apply (ErrorString &errorDescription)=0

Protected Member Functions

- **ILocalStoragePatch** (QObject *parent=nullptr)

Friends

- class **LocalStorageDatabaseUpgrader**

4.29.1 Detailed Description

The ILocalStoragePatch class represents the interface for patches of local storage. Each such patch somehow changes the layout of local storage persistence so that only compliant & corresponding versions of libquentier can be used to work with it.

4.29.2 Member Function Documentation

4.29.2.1 apply()

```
virtual bool quentier::ILocalStoragePatch::apply (
            ErrorString & errorDescription )  [pure virtual]
```

Apply the patch to local storage

Parameters

errorDescription	The textual description of the error in case of patch application failure

Returns

True in case of successful patch application, false otherwise

4.29.2.2 backupLocalStorage()

```
virtual bool quentier::ILocalStoragePatch::backupLocalStorage (
            ErrorString & errorDescription )  [pure virtual]
```

Backup either the entire local storage or its parts affected by the particular patch, should be called before applying the patch (but can be skipped if not desired).

Parameters

errorDescription	The textual description of the error in case of backup preparation failure

Returns

True if the local storage was backed up for the patch successfully, false otherwise

4.29.2.3 backupProgress

```
void quentier::ILocalStoragePatch::backupProgress (
            double progress )  [signal]
```

Local storage backup preparation progress

Parameters

progress	Backup preparation progress value, from 0 to 1

4.29.2.4 fromVersion()

```
virtual int quentier::ILocalStoragePatch::fromVersion ( ) const  [pure virtual]
```

Returns

Version of local storage to which the patch needs to be applied

4.29.2.5 patchLongDescription()

```
virtual QString quentier::ILocalStoragePatch::patchLongDescription ( ) const  [pure virtual]
```

Returns

Long i.e. detailed description of the patch

4.29.2.6 patchShortDescription()

```
virtual QString quentier::ILocalStoragePatch::patchShortDescription ( ) const  [pure virtual]
```

Returns

Short description of the patch

4.29.2.7 progress

```
void quentier::ILocalStoragePatch::progress (
            double progress )  [signal]
```

Patch application progress signal

Parameters

progress	Patch application progress value, from 0 to 1

4.29.2.8 removeLocalStorageBackup()

```
virtual bool quentier::ILocalStoragePatch::removeLocalStorageBackup (
            ErrorString & errorDescription )  [pure virtual]
```

Remove the previously made backup of local storage, presumably after successful application of the patch so the backup is no longer needed. It won't work if no backup was made before applying a patch, obviously.

Parameters

errorDescription	The textual description of the error in case of failure to remove the local storage backup

Returns

True if local storage backup was successfully removed, false otherwise

4.29.2.9 restoreBackupProgress

```
void quentier::ILocalStoragePatch::restoreBackupProgress (
            double progress )  [signal]
```

Local storage restoration from backup progress

Parameters

progress	Local storage restoration from backup progress value, from 0 to 1

4.29.2.10 restoreLocalStorageFromBackup()

```
virtual bool quentier::ILocalStoragePatch::restoreLocalStorageFromBackup (
            ErrorString & errorDescription )  [pure virtual]
```

Restore local storage from previously made backup, presumably after the failed attempt to apply a patch. Won't work if no backup was made before applying a patch, obviously.

Parameters

errorDescription	The textual description of the error in case of failure to restore the local storage from backup

Returns

True if local storage was successfully restored from backup, false otherwise

4.29.2.11 toVersion()

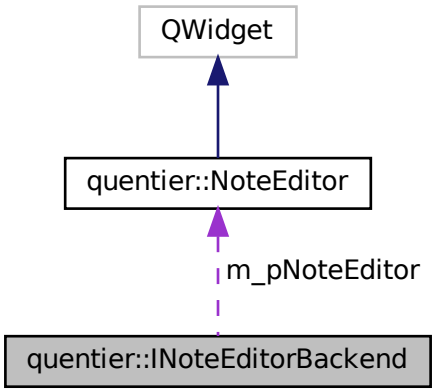
```
virtual int quentier::ILocalStoragePatch::toVersion ( ) const  [pure virtual]
```

Returns

Version of local storage to which the patch would upgrade the local storage

4.30 quentier::INoteEditorBackend Class Reference

Collaboration diagram for quentier::INoteEditorBackend:



Public Types

- enum **Rotation** { **Clockwise** = 0, **Counterclockwise** }

Public Member Functions

- virtual void **initialize** (LocalStorageManagerAsync &localStorageManager, SpellChecker &spellChecker, const Account &account, QThread *pBackgroundJobsThread)=0
- virtual QObject * **object** ()=0
- virtual QWidget * **widget** ()=0
- virtual void **setAccount** (const Account &account)=0
- virtual void **setUndoStack** (QUndoStack *pUndoStack)=0
- virtual void **setInitialPageHtml** (const QString &html)=0
- virtual void **setNoteNotFoundPageHtml** (const QString &html)=0
- virtual void **setNoteDeletedPageHtml** (const QString &html)=0
- virtual void **setNoteLoadingPageHtml** (const QString &html)=0
- virtual bool **isNoteLoaded** () const =0
- virtual qint64 **idleTime** () const =0
- virtual void **convertToNote** ()=0
- virtual void **saveNoteToLocalStorage** ()=0
- virtual void **setNoteTitle** (const QString &noteTitle)=0
- virtual void **setTagIds** (const QStringList &tagLocalUids, const QStringList &tagGuids)=0
- virtual void **undo** ()=0
- virtual void **redo** ()=0

- virtual void **cut** ()=0
- virtual void **copy** ()=0
- virtual void **paste** ()=0
- virtual void **pasteUnformatted** ()=0
- virtual void **selectAll** ()=0
- virtual void **formatSelectionAsSourceCode** ()=0
- virtual void **fontMenu** ()=0
- virtual void **textBold** ()=0
- virtual void **textItalic** ()=0
- virtual void **textUnderline** ()=0
- virtual void **textStrikethrough** ()=0
- virtual void **textHighlight** ()=0
- virtual void **alignLeft** ()=0
- virtual void **alignCenter** ()=0
- virtual void **alignRight** ()=0
- virtual void **alignFull** ()=0
- virtual QString **selectedText** () const =0
- virtual bool **hasSelection** () const =0
- virtual void **findNext** (const QString &text, const bool matchCase) const =0
- virtual void **findPrevious** (const QString &text, const bool matchCase) const =0
- virtual void **replace** (const QString &textToReplace, const QString &replacementText, const bool match↩ Case)=0
- virtual void **replaceAll** (const QString &textToReplace, const QString &replacementText, const bool match↩ Case)=0
- virtual void **insertToDoCheckbox** ()=0
- virtual void **insertInAppNoteLink** (const QString &userId, const QString &shardId, const QString &noteGuid, const QString &linkText)=0
- virtual void **setSpellcheck** (const bool enabled)=0
- virtual bool **spellCheckEnabled** () const =0
- virtual void **setFont** (const QFont &font)=0
- virtual void **setFontHeight** (const int height)=0
- virtual void **setFontColor** (const QColor &color)=0
- virtual void **setBackgroundColor** (const QColor &color)=0
- virtual QPalette **defaultPalette** () const =0
- virtual void **setDefaultPalette** (const QPalette &pal)=0
- virtual const QFont * **defaultFont** () const =0
- virtual void **setDefaultFont** (const QFont &font)=0
- virtual void **insertHorizontalLine** ()=0
- virtual void **increaseFontSize** ()=0
- virtual void **decreaseFontSize** ()=0
- virtual void **increaseIndentation** ()=0
- virtual void **decreaseIndentation** ()=0
- virtual void **insertBulletedList** ()=0
- virtual void **insertNumberedList** ()=0
- virtual void **insertTableDialog** ()=0
- virtual void **insertFixedWidthTable** (const int rows, const int columns, const int widthInPixels)=0
- virtual void **insertRelativeWidthTable** (const int rows, const int columns, const double relativeWidth)=0
- virtual void **insertTableRow** ()=0
- virtual void **insertTableColumn** ()=0
- virtual void **removeTableRow** ()=0
- virtual void **removeTableColumn** ()=0
- virtual void **addAttachmentDialog** ()=0
- virtual void **saveAttachmentDialog** (const QByteArray &resourceHash)=0
- virtual void **saveAttachmentUnderCursor** ()=0
- virtual void **openAttachment** (const QByteArray &resourceHash)=0

- virtual void **openAttachmentUnderCursor** ()=0
- virtual void **copyAttachment** (const QByteArray &resourceHash)=0
- virtual void **copyAttachmentUnderCursor** ()=0
- virtual void **removeAttachment** (const QByteArray &resourceHash)=0
- virtual void **removeAttachmentUnderCursor** ()=0
- virtual void **renameAttachment** (const QByteArray &resourceHash)=0
- virtual void **renameAttachmentUnderCursor** ()=0
- virtual void **rotateImageAttachment** (const QByteArray &resourceHash, const Rotation rotationDirection)=0
- virtual void **rotateImageAttachmentUnderCursor** (const Rotation rotationDirection)=0
- virtual void **encryptSelectedText** ()=0
- virtual void **decryptEncryptedTextUnderCursor** ()=0
- virtual void **decryptEncryptedText** (QString encryptedText, QString cipher, QString keyLength, QString hint, QString enCryptIndex)=0
- virtual void **hideDecryptedTextUnderCursor** ()=0
- virtual void **hideDecryptedText** (QString encryptedText, QString decryptedText, QString cipher, QString keyLength, QString hint, QString enDecryptedIndex)=0
- virtual void **editHyperlinkDialog** ()=0
- virtual void **copyHyperlink** ()=0
- virtual void **removeHyperlink** ()=0
- virtual void **onNoteLoadCancelled** ()=0
- virtual bool **print** (QPrinter &printer, ErrorString &errorDescription)=0
- virtual bool **exportToPdf** (const QString &absoluteFilePath, ErrorString &errorDescription)=0
- virtual bool **exportToEnex** (const QStringList &tagNames, QString &enex, ErrorString &errorDescription)=0
- virtual QString **currentNoteLocalUid** () const =0
- virtual void **setCurrentNoteLocalUid** (const QString &noteLocalUid)=0
- virtual void **clear** ()=0
- virtual bool **isModified** () const =0
- virtual bool **isEditorPageModified** () const =0
- virtual void **setFocusToEditor** ()=0

Protected Member Functions

- **INoteEditorBackend** (NoteEditor *parent)

Protected Attributes

- NoteEditor * **m_pNoteEditor**

Friends

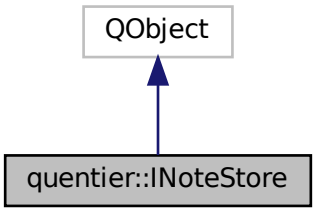
- QUENTIER_EXPORT QTextStream & **operator<<** (QTextStream &strm, const Rotation rotation)
- QUENTIER_EXPORT QDebug & **operator<<** (QDebug &dbg, const Rotation rotation)

4.31 quentier::INoteStore Class Reference

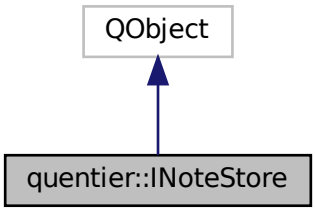
INoteStore is the interface which provides methods required for the implementation of NoteStore part of Evernote EDAM sync protocol.

`#include <INoteStore.h>`

Inheritance diagram for quentier::INoteStore:



Collaboration diagram for quentier::INoteStore:



Signals

- void **getNoteAsyncFinished** (qint32 errorCode, qevercloud::Note note, qint32 rateLimitSeconds, ErrorString errorDescription)
- void **getResourceAsyncFinished** (qint32 errorCode, qevercloud::Resource resource, qint32 rateLimit↩ Seconds, ErrorString errorDescription)

Public Member Functions

- virtual INoteStore ∗ **create** () const =0
- virtual QString noteStoreUrl () const =0
- virtual void setNoteStoreUrl (QString noteStoreUrl)=0

- virtual void setAuthData (QString authenticationToken, QList< QNetworkCookie > cookies)=0
- virtual void stop ()=0
- virtual qint32 createNotebook (Notebook &notebook, ErrorString &errorDescription, qint32 &rateLimit↩ Seconds, QString linkedNotebookAuthToken={})=0
- virtual qint32 updateNotebook (Notebook &notebook, ErrorString &errorDescription, qint32 &rateLimit↩ Seconds, QString linkedNotebookAuthToken={})=0
- virtual qint32 createNote (Note &note, ErrorString &errorDescription, qint32 &rateLimitSeconds, QString linkedNotebookAuthToken={})=0
- virtual qint32 updateNote (Note &note, ErrorString &errorDescription, qint32 &rateLimitSeconds, QString linkedNotebookAuthToken={})=0
- virtual qint32 createTag (Tag &tag, ErrorString &errorDescription, qint32 &rateLimitSeconds, QString linked↩ NotebookAuthToken={})=0
- virtual qint32 updateTag (Tag &tag, ErrorString &errorDescription, qint32 &rateLimitSeconds, QString linkedNotebookAuthToken={})=0
- virtual qint32 createSavedSearch (SavedSearch &savedSearch, ErrorString &errorDescription, qint32 &rateLimitSeconds)=0
- virtual qint32 updateSavedSearch (SavedSearch &savedSearch, ErrorString &errorDescription, qint32 &rateLimitSeconds)=0
- virtual qint32 getSyncState (qevercloud::SyncState &syncState, ErrorString &errorDescription, qint32 &rate↩ LimitSeconds)=0
- virtual qint32 getSyncChunk (const qint32 afterUSN, const qint32 maxEntries, const qevercloud::Sync↩ ChunkFilter &filter, qevercloud::SyncChunk &syncChunk, ErrorString &errorDescription, qint32 &rateLimit↩ Seconds)=0
- virtual qint32 getLinkedNotebookSyncState (const qevercloud::LinkedNotebook &linkedNotebook, const Q↩ String &authToken, qevercloud::SyncState &syncState, ErrorString &errorDescription, qint32 &rateLimit↩ Seconds)=0
- virtual qint32 getLinkedNotebookSyncChunk (const qevercloud::LinkedNotebook &linkedNotebook, const qint32 afterUSN, const qint32 maxEntries, const QString &linkedNotebookAuthToken, const bool fullSync↩ Only, qevercloud::SyncChunk &syncChunk, ErrorString &errorDescription, qint32 &rateLimitSeconds)=0
- virtual qint32 getNote (const bool withContent, const bool withResourcesData, const bool withResources↩ Recognition, const bool withResourceAlternateData, Note &note, ErrorString &errorDescription, qint32 &rateLimitSeconds)=0
- virtual bool getNoteAsync (const bool withContent, const bool withResourceData, const bool with↩ ResourcesRecognition, const bool withResourceAlternateData, const bool withSharedNotes, const bool withNoteAppDataValues, const bool withResourceAppDataValues, const bool withNoteLimits, const QString &noteGuid, const QString &authToken, ErrorString &errorDescription)=0
- virtual qint32 getResource (const bool withDataBody, const bool withRecognitionDataBody, const bool with↩ AlternateDataBody, const bool withAttributes, const QString &authToken, Resource &resource, ErrorString &errorDescription, qint32 &rateLimitSeconds)=0
- virtual bool getResourceAsync (const bool withDataBody, const bool withRecognitionDataBody, const bool withAlternateDataBody, const bool withAttributes, const QString &resourceGuid, const QString &authToken, ErrorString &errorDescription)=0
- virtual qint32 authenticateToSharedNotebook (const QString &shareKey, qevercloud::AuthenticationResult &authResult, ErrorString &errorDescription, qint32 &rateLimitSeconds)=0

Protected Member Functions

- **INoteStore** (QObject *parent=nullptr)

4.31.1 Detailed Description

INoteStore is the interface which provides methods required for the implementation of NoteStore part of Evernote EDAM sync protocol.

4.31.2 Member Function Documentation

4.31.2.1 authenticateToSharedNotebook()

```
virtual qint32 quentier::INoteStore::authenticateToSharedNotebook (
            const QString & shareKey,
            qevercloud::AuthenticationResult & authResult,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds )  [pure virtual]
```

Authenticate to shared notebook

Parameters

shareKey	The shared notebook global identifier
authResult	Output parameter, the result of authentication
errorDescription	The textual description of the error if authentication to shared notebook could not be performed
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful authentication to shared notebook, other values corresponding to qevercloud::EDAMErrorCode enumeration instead

4.31.2.2 createNote()

```
virtual qint32 quentier::INoteStore::createNote (
            Note & note,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds,
            QString linkedNotebookAuthToken = {} )  [pure virtual]
```

Create note

Parameters

note	Note to be created
errorDescription	The textual description of the error in case note could not be created
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED
linkedNotebookAuthToken	If a note is created within another user's account, the corresponding auth token should be set, otherwise the note would be created in user's own account

Returns

Error code, 0 in case of successful note creation, other values corresponding to qevercloud::EDAMErrorCode enumeration instead

4.31.2.3 createNotebook()

```
virtual qint32 quentier::INoteStore::createNotebook (
            Notebook & notebook,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds,
            QString linkedNotebookAuthToken = {} )  [pure virtual]
```

Create notebook

Parameters

notebook	Notebook to be created, must have name set and either "active" or "default notebook" fields may be set
errorDescription	The textual description of the error in case notebook could not be created
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED
linkedNotebookAuthToken	If a notebook is created within another user's account, the corresponding auth token should be set, otherwise the notebook would be created in user's own account

Returns

Error code, 0 in case of successful notebook creation, other values corresponding to qevercloud::EDAM↩ ErrorCode enumeration instead

4.31.2.4 createSavedSearch()

```
virtual qint32 quentier::INoteStore::createSavedSearch (
            SavedSearch & savedSearch,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds )  [pure virtual]
```

Create saved search

Parameters

savedSearch	Saved search to be created, must have name and query set, can also have search scope set
errorDescription	The textual description of the error in case saved search could not be created
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned
	value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful saved search creation, other values corresponding to qevercloud::EDAMErrorCode enumeration instead

4.31.2.5 createTag()

```
virtual qint32 quentier::INoteStore::createTag (
            Tag & tag,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds,
            QString linkedNotebookAuthToken = {} )  [pure virtual]
```

Create tag

Parameters

note	Tag to be created, must have name set, can also have parent guid set
errorDescription	The textual description of the error in case tag could not be created
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED
linkedNotebookAuthToken	If a tag is created within another user's account, the corresponding auth token should be set, otherwise the tag would be created in user's own account

Returns

Error code, 0 in case of successful tag creation, other values corresponding to qevercloud::EDAMErrorCode enumeration instead

4.31.2.6 getLinkedNotebookSyncChunk()

```
virtual qint32 quentier::INoteStore::getLinkedNotebookSyncChunk (
            const qevercloud::LinkedNotebook & linkedNotebook,
            const qint32 afterUSN,
            const qint32 maxEntries,
            const QString & linkedNotebookAuthToken,
            const bool fullSyncOnly,
            qevercloud::SyncChunk & syncChunk,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds )  [pure virtual]
```

Get linked notebook sync chunk

Parameters

linkedNotebook	The linked notebook for which the sync chunk is being retrieved, must contain identifying information and permissions to access the notebook in question
afterUSN	The USN after which the sync chunks are being requested
maxEntries	Max number of items within the sync chunk to be returned
linkedNotebookAuthToken	The authentication token to use for the data from the linked notebook
fullSyncOnly	If true then client only wants initial data for a full sync. In this case Evernote service will not return any expunged objects and will not return any resources since these are also provided in their corresponding notes
syncChunk	Output parameter, the sync chunk
errorDescription	The textual description of the error in case linked notebook sync chunk could not be retrieved
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful linked notebook sync chunk retrieval, other values corresponding to qevercloud::EDAMErrorCode enumeration instead

4.31.2.7 getLinkedNotebookSyncState()

```
virtual qint32 quentier::INoteStore::getLinkedNotebookSyncState (
             const qevercloud::LinkedNotebook & linkedNotebook,
             const QString & authToken,
             qevercloud::SyncState & syncState,
             ErrorString & errorDescription,
             qint32 & rateLimitSeconds )  [pure virtual]
```

Get linked notebook sync state

Parameters

linkedNotebook	The linked notebook for which the sync state is being retrieved, must contain identifying information and permissions to access the notebook in question
authToken	The authentication token to use for the data from the linked notebook
syncState	Output parameter, the sync state
errorDescription	The textual description of the error in case linked notebook sync state could not be retrieved
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful linked notebook sync state retrieval, other values corresponding to qevercloud::EDAMErrorCode enumeration instead

4.31.2.8 getNote()

```
virtual qint32 quentier::INoteStore::getNote (
            const bool withContent,
            const bool withResourcesData,
            const bool withResourcesRecognition,
            const bool withResourceAlternateData,
            Note & note,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds )  [pure virtual]
```

Get note synchronously

Parameters

withContent	If true, the returned note would include the content
withResourcesData	If true, any resources the note might have would include their full data
withResourcesRecognition	If true, any resources the note might have and which have Evernote supplied recognition would include their full recognition data
withResourceAlternateData	If true, any resources the note might have would include their full alternate data
note	Input and output parameter, the retrieved note, must have guid set
errorDescription	The textual description of the error in case the note could not be retrieved
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful note retrieval, other values corresponding to qevercloud::EDAMErrorCode enumeration instead

4.31.2.9 getNoteAsync()

```
virtual bool quentier::INoteStore::getNoteAsync (
            const bool withContent,
            const bool withResourceData,
            const bool withResourcesRecognition,
            const bool withResourceAlternateData,
            const bool withSharedNotes,
            const bool withNoteAppDataValues,
            const bool withResourceAppDataValues,
```

```
            const bool withNoteLimits,
            const QString & noteGuid,
            const QString & authToken,
            ErrorString & errorDescription )  [pure virtual]
```

Get note asynchronously

If the method returned true, the actual result of the method invokation would be returned via the emission of getNoteAsyncFinished signal.

Parameters

withContent	If true, the returned note would include the content
withResourcesData	If true, any resources the note might have would include their full data
withResourcesRecognition	If true, any resources the note might have and which have Evernote supplied recognition would include their full recognition data
withResourceAlternateData	If true, any resources the note might have would include their full alternate data
withSharedNotes	If true, any shared notes contained within the note would be provided along with the asynchronously fetched result
withNoteAppDataValues	If true, the asynchronously fetched note would contain the app data values
withResourceAppDataValues	If true, the resources of asynchronously fetched note would contain the app data values
withNoteLimits	If true, the asynchronously fetched note would contain note limits
noteGuid	The guid of the note to be retrieved
authToken	Authentication token to use for note retrieval
errorDescription	The textual description of the error if the launch of async note retrieval has failed

Returns

True if the launch of async note retrieval was successful, false otherwise

4.31.2.10 getResource()

```
virtual qint32 quentier::INoteStore::getResource (
            const bool withDataBody,
            const bool withRecognitionDataBody,
            const bool withAlternateDataBody,
            const bool withAttributes,
            const QString & authToken,
            Resource & resource,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds )  [pure virtual]
```

Get resource synchronously

Parameters

withDataBody	If true, the returned resource would include its data body
withRecognitionDataBody	If true, the returned resource would include its recognition data body

Parameters

withAlternateDataBody	If true, the returned resource would include its alternate data body
withAttributes	If true, the returned resource would include its attributes
authToken	Authentication token to use for resource retrieval
resource	Input and output parameter, the retrieved resource, must have guid set
errorDescription	The textual description of the error in case the resource could not be retrieved
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful resource retrieval, other values corresponding to qevercloud::EDAM↩ ErrorCode enumeration instead

4.31.2.11 getResourceAsync()

```
virtual bool quentier::INoteStore::getResourceAsync (
             const bool withDataBody,
             const bool withRecognitionDataBody,
             const bool withAlternateDataBody,
             const bool withAttributes,
             const QString & resourceGuid,
             const QString & authToken,
             ErrorString & errorDescription )  [pure virtual]
```

Get resource asynchronously

If the method returned true, the actual result of the method invokation would be returned via the emission of get↩ ResourceAsyncFinished signal.

Parameters

withDataBody	If true, the returned resource would include its data body
withRecognitionDataBody	If true, the returned resource would include its recognition data body
withAlternateDataBody	If true, the returned resource would include its alternate data body
withAttributes	If true, the returned resource would include its attributes
resourceGuid	The guid of the resource to be retrieved
authToken	Authentication token to use for resource retrieval
errorDescription	The textual description of the error if the launch of async resource retrieval has failed

Returns

True if the launch of async resource retrieval was successful, false otherwise

4.31.2.12 getSyncChunk()

```
virtual qint32 quentier::INoteStore::getSyncChunk (
            const qint32 afterUSN,
            const qint32 maxEntries,
            const qevercloud::SyncChunkFilter & filter,
            qevercloud::SyncChunk & syncChunk,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds )  [pure virtual]
```

Get sync chunk

Parameters

afterUSN	The USN after which the sync chunks are being requested
maxEntries	Max number of items within the sync chunk to be returned
filter	Filter for items to be returned within the sync chunks
syncChunk	Output parameter, the sync chunk
errorDescription	The textual description of the error in case sync chunk could not be retrieved
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful sync chunk retrieval, other values corresponding to qevercloud::EDAM↩ ErrorCode enumeration instead

4.31.2.13 getSyncState()

```
virtual qint32 quentier::INoteStore::getSyncState (
            qevercloud::SyncState & syncState,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds )  [pure virtual]
```

Get sync state

Parameters

syncState	Output parameter, the sync state
errorDescription	The textual description of the error in case sync state could not be retrieved
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful sync state retrieval, other values corresponding to qevercloud::EDAM↩ ErrorCode enumeration instead

4.31.2.14 noteStoreUrl()

```
virtual QString quentier::INoteStore::noteStoreUrl ( ) const  [pure virtual]
```

Provide note store URL used by this INoteStore instance

4.31.2.15 setAuthData()

```
virtual void quentier::INoteStore::setAuthData (
            QString authenticationToken,
            QList< QNetworkCookie > cookies )  [pure virtual]
```

Set authentication data to be used by this INoteStore instance

4.31.2.16 setNoteStoreUrl()

```
virtual void quentier::INoteStore::setNoteStoreUrl (
            QString noteStoreUrl )  [pure virtual]
```

Set note store URL to be used by this INoteStore instance

4.31.2.17 stop()

```
virtual void quentier::INoteStore::stop ( )  [pure virtual]
```

Stop asynchronous queries for notes or resources which might be running at the moment

4.31.2.18 updateNote()

```
virtual qint32 quentier::INoteStore::updateNote (
            Note & note,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds,
            QString linkedNotebookAuthToken = {} )  [pure virtual]
```

Update note

Parameters

note	Note to be updated, must have guid set
errorDescription	The textual description of the error in case note could not be updated
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED
linkedNotebookAuthToken	If a note is updated within another user's account, the corresponding auth token should be set, otherwise the note would be updated within user's own account

Returns

Error code, 0 in case of successful note update, other values corresponding to qevercloud::EDAMErrorCode enumeration instead

4.31.2.19 updateNotebook()

```
virtual qint32 quentier::INoteStore::updateNotebook (
            Notebook & notebook,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds,
            QString linkedNotebookAuthToken = {} )  [pure virtual]
```

Update notebook

Parameters

notebook	Notebook to be updated, must have guid set
errorDescription	The textual description of the error in case notebook could not be updated
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED
linkedNotebookAuthToken	If a notebook is updated within another user's account, the corresponding auth token should be set, otherwise the notebook would be updated within user's own account

Returns

Error code, 0 in case of successful notebook update, other values corresponding to qevercloud::EDAMError↩ Code enumeration instead

4.31.2.20 updateSavedSearch()

```
virtual qint32 quentier::INoteStore::updateSavedSearch (
            SavedSearch & savedSearch,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds )  [pure virtual]
```

Update saved search

Parameters

savedSearch	Saved search to be updated, must have guid set
errorDescription	The textual description of the error in case saved search could not be updated
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful saved search update, other values corresponding to qevercloud::EDAM↩ ErrorCode enumeration instead

4.31.2.21 updateTag()

```
virtual qint32 quentier::INoteStore::updateTag (
            Tag & tag,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds,
            QString linkedNotebookAuthToken = {} )  [pure virtual]
```

Update tag

Parameters

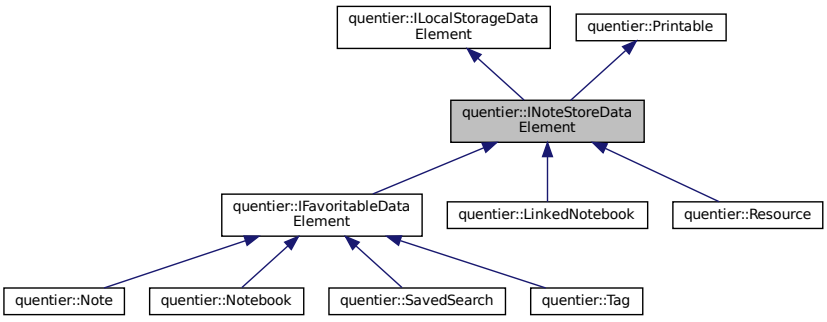
tag	Tag to be updated, must have guid set
errorDescription	The textual description of the error in case tag could not be updated
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED
linkedNotebookAuthToken	If a tag is updated within another user's account, the corresponding auth token should be set, otherwise the tag would be updated within user's own account

Returns

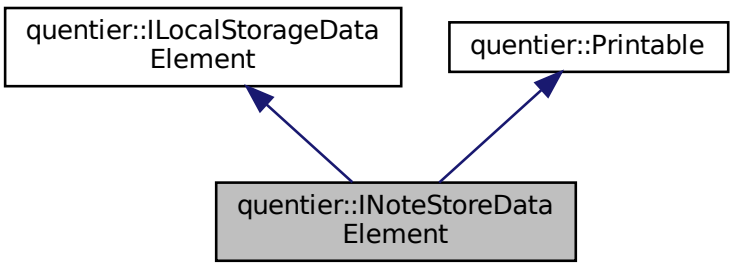
Error code, 0 in case of successful tag update, other values corresponding to qevercloud::EDAMErrorCode enumeration instead

4.32 quentier::INoteStoreDataElement Class Reference

Inheritance diagram for quentier::INoteStoreDataElement:



Collaboration diagram for quentier::INoteStoreDataElement:



Public Member Functions

- virtual void **clear** ()=0
- virtual bool **hasGuid** () const =0
- virtual const QString & **guid** () const =0
- virtual void **setGuid** (const QString &guid)=0
- virtual bool **hasUpdateSequenceNumber** () const =0
- virtual qint32 **updateSequenceNumber** () const =0
- virtual void **setUpdateSequenceNumber** (const qint32 usn)=0
- virtual bool **checkParameters** (ErrorString &errorDescription) const =0
- virtual bool **isDirty** () const =0
- virtual void **setDirty** (const bool dirty)=0
- virtual bool **isLocal** () const =0
- virtual void **setLocal** (const bool local)=0

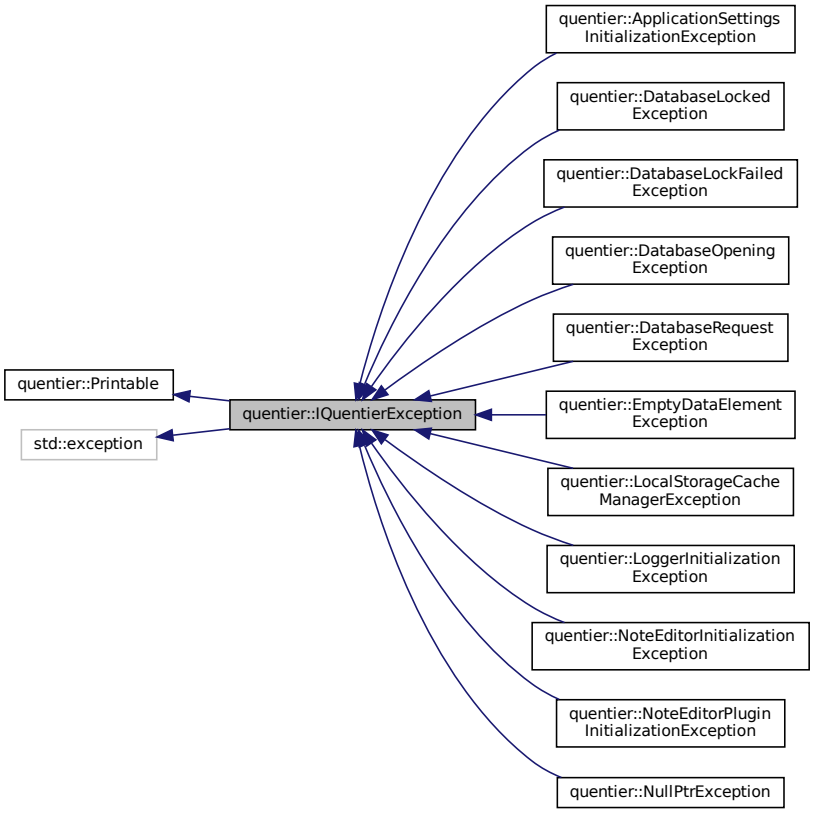
Additional Inherited Members

4.33 quentier::IQuentierException Class Reference

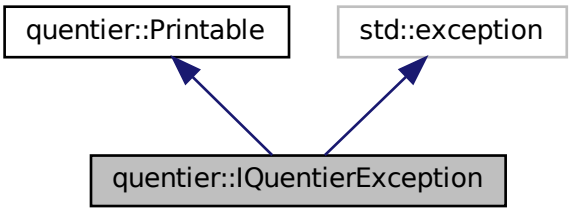
The IQuentierException class represents the interface for exceptions specific to libquentier and applications based on it.

`#include <IQuentierException.h>`

Inheritance diagram for quentier::IQuentierException:



Collaboration diagram for quentier::IQuentierException:



Public Member Functions

- **IQuentierException** (const ErrorString &message)
- QString **localizedErrorMessage** () const
- QString **nonLocalizedErrorMessage** () const
- virtual const char * **what** () const noexcept override
- virtual QTextStream & **print** (QTextStream &strm) const override

Protected Member Functions

- **IQuentierException** (const IQuentierException &other)
- IQuentierException & **operator=** (const IQuentierException &other)
- virtual const QString **exceptionDisplayName** () const =0

4.33.1 Detailed Description

The IQuentierException class represents the interface for exceptions specific to libquentier and applications based on it.

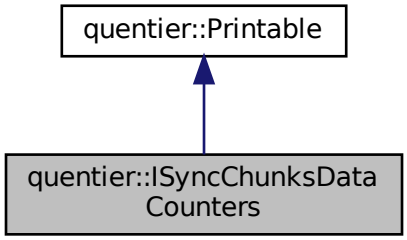
In addition to standard exception features inherited from std::exception, IQuentierException based exceptions can provide both localized and non-localized error messages.

4.34 quentier::ISyncChunksDataCounters Struct Reference

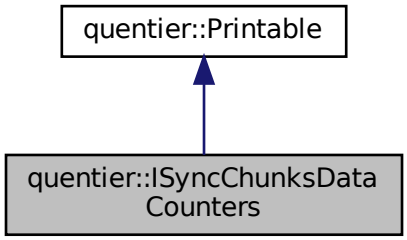
The ISyncChunksDataCounters interface provides integer counters representing the current progress on processing the data from downloaded sync chunks.

`#include <ISyncChunksDataCounters.h>`

Inheritance diagram for quentier::ISyncChunksDataCounters:



Collaboration diagram for quentier::ISyncChunksDataCounters:



Public Member Functions

- virtual quint64 totalSavedSearches () const noexcept=0
- virtual quint64 totalExpungedSavedSearches () const noexcept=0
- virtual quint64 addedSavedSearches () const noexcept=0
- virtual quint64 updatedSavedSearches () const noexcept=0
- virtual quint64 expungedSavedSearches () const noexcept=0
- virtual quint64 totalTags () const noexcept=0
- virtual quint64 totalExpungedTags () const noexcept=0
- virtual quint64 addedTags () const noexcept=0
- virtual quint64 updatedTags () const noexcept=0
- virtual quint64 expungedTags () const noexcept=0
- virtual quint64 totalLinkedNotebooks () const noexcept=0
- virtual quint64 totalExpungedLinkedNotebooks () const noexcept=0
- virtual quint64 addedLinkedNotebooks () const noexcept=0
- virtual quint64 updatedLinkedNotebooks () const noexcept=0
- virtual quint64 expungedLinkedNotebooks () const noexcept=0
- virtual quint64 totalNotebooks () const noexcept=0
- virtual quint64 totalExpungedNotebooks () const noexcept=0
- virtual quint64 addedNotebooks () const noexcept=0
- virtual quint64 updatedNotebooks () const noexcept=0
- virtual quint64 expungedNotebooks () const noexcept=0

Additional Inherited Members

4.34.1 Detailed Description

The ISyncChunksDataCounters interface provides integer counters representing the current progress on processing the data from downloaded sync chunks.

4.34.2 Member Function Documentation

4.34.2.1 addedLinkedNotebooks()

```
virtual quint64 quentier::ISyncChunksDataCounters::addedLinkedNotebooks ( ) const  [pure virtual],
[noexcept]
```

Number of linked notebooks from sync chunks added to the local storage so far

4.34.2.2 addedNotebooks()

```
virtual quint64 quentier::ISyncChunksDataCounters::addedNotebooks ( ) const  [pure virtual],
[noexcept]
```

Number of notebooks from sync chunks added to the local storage so far

4.34.2.3 addedSavedSearches()

```
virtual quint64 quentier::ISyncChunksDataCounters::addedSavedSearches ( ) const  [pure virtual],
[noexcept]
```

Number of saved searches from sync chunks added to the local storage so far

4.34.2.4 addedTags()

```
virtual quint64 quentier::ISyncChunksDataCounters::addedTags ( ) const  [pure virtual], [noexcept]
```

Number of tags from sync chunks added to the local storage so far

4.34.2.5 expungedLinkedNotebooks()

```
virtual quint64 quentier::ISyncChunksDataCounters::expungedLinkedNotebooks ( ) const  [pure
virtual], [noexcept]
```

Number of linked notebooks from sync chunks expunged from the local storage so far

4.34.2.6 expungedNotebooks()

```
virtual quint64 quentier::ISyncChunksDataCounters::expungedNotebooks ( ) const  [pure virtual],
[noexcept]
```

Number of notebooks from sync chunks expunged from the local storage so far

4.34.2.7 expungedSavedSearches()

```
virtual quint64 quentier::ISyncChunksDataCounters::expungedSavedSearches ( ) const  [pure
virtual], [noexcept]
```

Number of saved searches from sync chunks expunged from the local storage so far

4.34.2.8 expungedTags()

```
virtual quint64 quentier::ISyncChunksDataCounters::expungedTags ( ) const  [pure virtual],
[noexcept]
```

Number of tags from sync chunks expunged from the local storage so far

4.34.2.9 totalExpungedLinkedNotebooks()

```
virtual quint64 quentier::ISyncChunksDataCounters::totalExpungedLinkedNotebooks ( ) const
[pure virtual], [noexcept]
```

Total number of expunged saved searches in downloaded sync chunks

4.34.2.10 totalExpungedNotebooks()

```
virtual quint64 quentier::ISyncChunksDataCounters::totalExpungedNotebooks ( ) const  [pure
virtual], [noexcept]
```

Total number of expunged notebooks in downloaded sync chunks

4.34.2.11 totalExpungedSavedSearches()

```
virtual quint64 quentier::ISyncChunksDataCounters::totalExpungedSavedSearches ( ) const  [pure
virtual], [noexcept]
```

Total number of expunged saved searches in downloaded sync chunks

4.34.2.12 totalExpungedTags()

```
virtual quint64 quentier::ISyncChunksDataCounters::totalExpungedTags ( ) const  [pure virtual],
[noexcept]
```

Total number of expunged tags in downloaded sync chunks

4.34.2.13 totalLinkedNotebooks()

```
virtual quint64 quentier::ISyncChunksDataCounters::totalLinkedNotebooks ( ) const  [pure virtual],
[noexcept]
```

Total number of new or updated linked notebooks in downloaded sync chunks

4.34.2.14 totalNotebooks()

```
virtual quint64 quentier::ISyncChunksDataCounters::totalNotebooks ( ) const  [pure virtual],
[noexcept]
```

Total number of new or updated notebooks in downloaded sync chunks

4.34.2.15 totalSavedSearches()

```
virtual quint64 quentier::ISyncChunksDataCounters::totalSavedSearches ( ) const  [pure virtual],
[noexcept]
```

Total number of new or updated saved searches in downloaded sync chunks

4.34.2.16 totalTags()

```
virtual quint64 quentier::ISyncChunksDataCounters::totalTags ( ) const  [pure virtual], [noexcept]
```

Total number of new or updated tags in downloaded sync chunks

4.34.2.17 updatedLinkedNotebooks()

```
virtual quint64 quentier::ISyncChunksDataCounters::updatedLinkedNotebooks ( ) const  [pure
virtual], [noexcept]
```

Number of linked notebooks from sync chunks updated in the local storage so far

4.34.2.18 updatedNotebooks()

```
virtual quint64 quentier::ISyncChunksDataCounters::updatedNotebooks ( ) const  [pure virtual],
[noexcept]
```

Number of notebooks from sync chunks updated in the local storage so far

4.34.2.19 updatedSavedSearches()

```
virtual quint64 quentier::ISyncChunksDataCounters::updatedSavedSearches ( ) const  [pure virtual],
[noexcept]
```

Number of saved searches from sync chunks updated in the local storage so far

4.34.2.20 updatedTags()

```
virtual quint64 quentier::ISyncChunksDataCounters::updatedTags ( ) const  [pure virtual],
[noexcept]
```

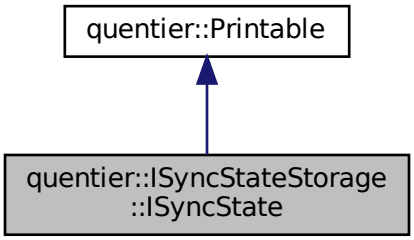
Number of tags from sync chunks updated in the local storage so far

4.35 quentier::ISyncStateStorage::ISyncState Class Reference

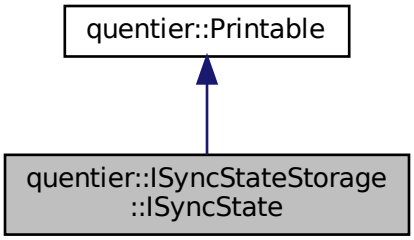
The ISyncState interface provides accessory methods to determine the sync state for the account.

```
#include <ISyncStateStorage.h>
```

Inheritance diagram for quentier::ISyncStateStorage::ISyncState:



Collaboration diagram for quentier::ISyncStateStorage::ISyncState:



Public Member Functions

- virtual qint32 **userDataUpdateCount** () const =0
- virtual qevercloud::Timestamp **userDataLastSyncTime** () const =0
- virtual QHash< QString, qint32 > **linkedNotebookUpdateCounts** () const =0
- virtual QHash< QString, qevercloud::Timestamp > **linkedNotebookLastSyncTimes** () const =0
- virtual QTextStream & **print** (QTextStream &strm) const override

Additional Inherited Members

4.35.1 Detailed Description

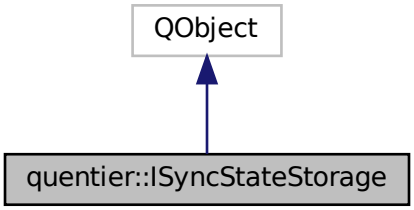
The ISyncState interface provides accessory methods to determine the sync state for the account.

4.36 quentier::ISyncStateStorage Class Reference

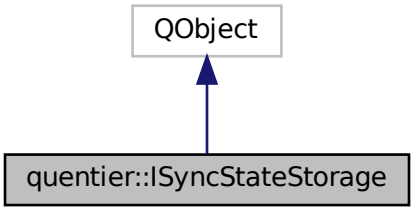
The ISyncStateStorage interface represents the interface of a class which stores sync state for given accounts persistently and provides access to previously stores sync states.

`#include <ISyncStateStorage.h>`

Inheritance diagram for quentier::ISyncStateStorage:



Collaboration diagram for quentier::ISyncStateStorage:



Classes

- class ISyncState

  The ISyncState interface provides accessory methods to determine the sync state for the account.

Public Types

- using **ISyncStatePtr** = std::shared_ptr< ISyncState >

Signals

- void notifySyncStateUpdated (Account account, ISyncStatePtr syncState)

Public Member Functions

- **ISyncStateStorage** (QObject *parent=nullptr)
- virtual ISyncStatePtr **getSyncState** (const Account &account)=0
- virtual void **setSyncState** (const Account &account, ISyncStatePtr syncState)=0

4.36.1 Detailed Description

The ISyncStateStorage interface represents the interface of a class which stores sync state for given accounts persistently and provides access to previously stores sync states.

4.36.2 Member Function Documentation

4.36.2.1 notifySyncStateUpdated

```
void quentier::ISyncStateStorage::notifySyncStateUpdated (
            Account account,
            ISyncStatePtr syncState )  [signal]
```

Classes implementing ISyncStateStorage interface are expected to emit notifySyncStateUpdated signal each time when sync state for the corresponding account is updated

4.37 quentier::IUserStore Class Reference

IUserStore is the interface which provides methods required for the implementation of UserStore part of Evernote EDAM sync protocol.

```
#include <IUserStore.h>
```

Public Member Functions

- virtual void setAuthData (QString authenticationToken, QList< QNetworkCookie > cookies)=0
- virtual bool checkVersion (const QString &clientName, qint16 edamVersionMajor, qint16 edamVersionMinor, ErrorString &errorDescription)=0
- virtual qint32 getUser (User &user, ErrorString &errorDescription, qint32 &rateLimitSeconds)=0
- virtual qint32 getAccountLimits (const qevercloud::ServiceLevel serviceLevel, qevercloud::AccountLimits &limits, ErrorString &errorDescription, qint32 &rateLimitSeconds)=0

4.37.1 Detailed Description

IUserStore is the interface which provides methods required for the implementation of UserStore part of Evernote EDAM sync protocol.

4.37.2 Member Function Documentation

4.37.2.1 checkVersion()

```
virtual bool quentier::IUserStore::checkVersion (
            const QString & clientName,
            qint16 edamVersionMajor,
            qint16 edamVersionMinor,
            ErrorString & errorDescription )  [pure virtual]
```

Check the version of EDAM protocol

Parameters

clientName	Application name + application version + platform name string
edamVersionMajor	The major version of EDAM protocol the application wants to use to connect to Evernote
edamVersionMinor	The minor version of EDAM protocol the application wants to use to connect to Evernote
errorDescription	The textual description of the error if the supplied protocol version cannot be used to connect to Evernote

Returns

True if protocol check was successful i.e. the service can talk to the client using the supplied protocol version, false otherwise

4.37.2.2 getAccountLimits()

```
virtual qint32 quentier::IUserStore::getAccountLimits (
            const qevercloud::ServiceLevel serviceLevel,
            qevercloud::AccountLimits & limits,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds )  [pure virtual]
```

Retrieve account limits corresponding to certain provided service level

Parameters

serviceLevel	The level of Evernote service for which account limits are requested
limits	Output account limits
errorDescription	The textual description of the error if account limits could not be retrieved
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful retrieval of account limits for the given service level, other values corresponding to qevercoud::EDAMErrorCode::type enumeration instead

4.37.2.3 getUser()

```
virtual qint32 quentier::IUserStore::getUser (
            User & user,
            ErrorString & errorDescription,
            qint32 & rateLimitSeconds )  [pure virtual]
```

Retrieve full information about user (account)

Parameters

user	Input and output parameter; on input needs to have user id set
errorDescription	The textual description of the error if full user information could not be retrieved
rateLimitSeconds	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED

Returns

Error code, 0 in case of successful retrieval of full user information, other values corresponding to qevercloud↩ ::EDAMErrorCode enumeration instead

4.37.2.4 setAuthData()

```
virtual void quentier::IUserStore::setAuthData (
            QString authenticationToken,
            QList< QNetworkCookie > cookies )  [pure virtual]
```

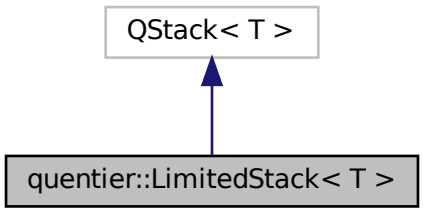
Set authentication data to be used by this IUserStore instance

4.38 quentier::LimitedStack< T > Class Template Reference

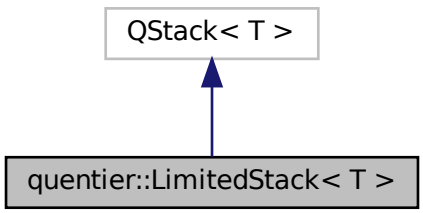
The LimitedStack template class implements a stack which may have a limitation for its size; when the size becomes too much according to the limit, the bottom element of the stack gets erased from it. Only limits greater than zero are considered.

`#include <LimitedStack.h>`

Inheritance diagram for quentier::LimitedStack< T >:



Collaboration diagram for quentier::LimitedStack< T >:



Public Member Functions

- **LimitedStack** (void(∗deleter)(T &)=nullptr)
- **LimitedStack** (const LimitedStack< T > &other)
- **LimitedStack** (LimitedStack< T > &&other)
- LimitedStack< T > & **operator=** (const LimitedStack< T > &other)
- LimitedStack< T > & **operator=** (LimitedStack< T > &&other)
- void **swap** (const LimitedStack< T > &other)
- int **limit** () const
- void **setLimit** (const int limit)
- void **push** (const T &t)

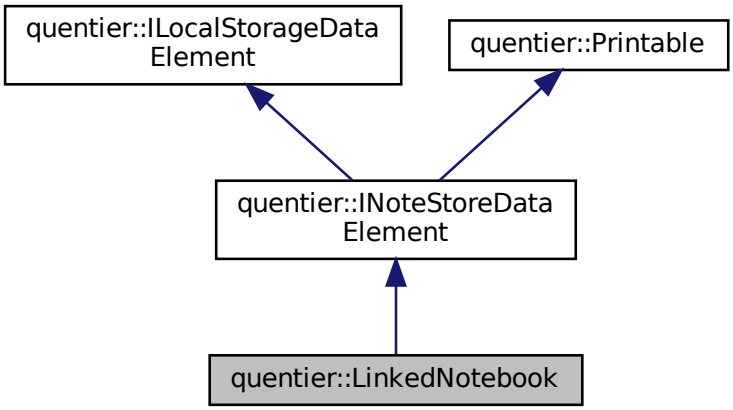
4.38.1 Detailed Description

template<class T>
class quentier::LimitedStack< T >

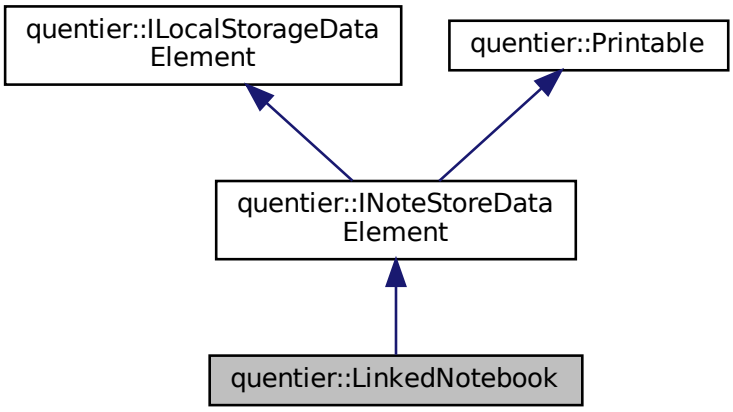
The LimitedStack template class implements a stack which may have a limitation for its size; when the size becomes too much according to the limit, the bottom element of the stack gets erased from it. Only limits greater than zero are considered.

4.39 quentier::LinkedNotebook Class Reference

Inheritance diagram for quentier::LinkedNotebook:



Collaboration diagram for quentier::LinkedNotebook:



Public Member Functions

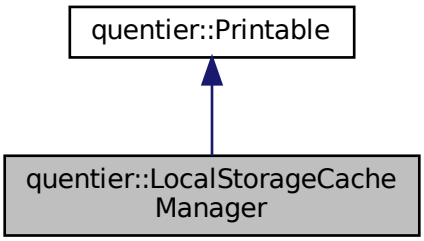
- **LinkedNotebook** (const LinkedNotebook &other)
- **LinkedNotebook** (LinkedNotebook &&other)
- LinkedNotebook & **operator=** (const LinkedNotebook &other)
- LinkedNotebook & **operator=** (LinkedNotebook &&other)
- **LinkedNotebook** (const qevercloud::LinkedNotebook &linkedNotebook)
- **LinkedNotebook** (qevercloud::LinkedNotebook &&linkedNotebook)
- const qevercloud::LinkedNotebook & **qevercloudLinkedNotebook** () const
- qevercloud::LinkedNotebook & **qevercloudLinkedNotebook** ()
- bool **operator==** (const LinkedNotebook &other) const
- bool **operator!=** (const LinkedNotebook &other) const
- virtual void **clear** () override
- virtual bool **hasGuid** () const override
- virtual const QString & **guid** () const override
- virtual void **setGuid** (const QString &guid) override
- virtual bool **hasUpdateSequenceNumber** () const override
- virtual qint32 **updateSequenceNumber** () const override
- virtual void **setUpdateSequenceNumber** (const qint32 usn) override
- virtual bool **checkParameters** (ErrorString &errorDescription) const override
- bool **hasShareName** () const
- const QString & **shareName** () const
- void **setShareName** (const QString &shareName)
- bool **hasUsername** () const
- const QString & **username** () const
- void **setUsername** (const QString &username)
- bool **hasShardId** () const
- const QString & **shardId** () const
- void **setShardId** (const QString &shardId)
- bool **hasSharedNotebookGlobalId** () const
- const QString & **sharedNotebookGlobalId** () const
- void **setSharedNotebookGlobalId** (const QString &sharedNotebookGlobalId)

- bool **hasUri** () const
- const QString & **uri** () const
- void **setUri** (const QString &uri)
- bool **hasNoteStoreUrl** () const
- const QString & **noteStoreUrl** () const
- void **setNoteStoreUrl** (const QString &noteStoreUrl)
- bool **hasWebApiUrlPrefix** () const
- const QString & **webApiUrlPrefix** () const
- void **setWebApiUrlPrefix** (const QString &webApiUrlPrefix)
- bool **hasStack** () const
- const QString & **stack** () const
- void **setStack** (const QString &stack)
- bool **hasBusinessId** () const
- qint32 **businessId** () const
- void **setBusinessId** (const qint32 businessId)
- virtual QTextStream & **print** (QTextStream &strm) const override

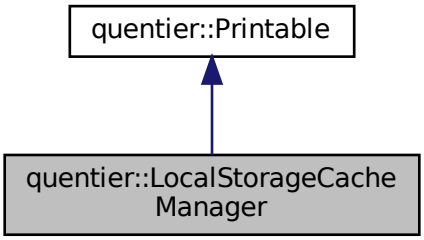
Additional Inherited Members

4.40 quentier::LocalStorageCacheManager Class Reference

Inheritance diagram for quentier::LocalStorageCacheManager:



Collaboration diagram for quentier::LocalStorageCacheManager:



Public Types

- enum **WhichUid** { **LocalUid**, **Guid** }

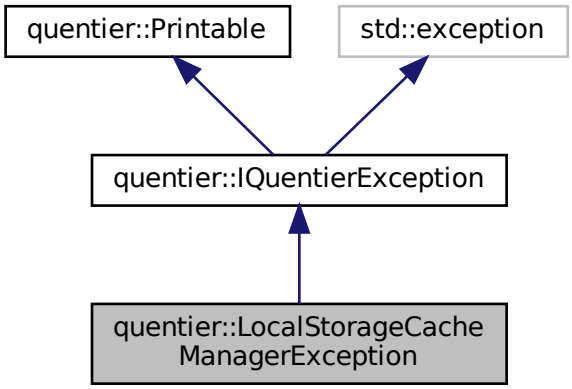
Public Member Functions

- void **clear** ()
- bool **empty** () const
- size_t **numCachedNotes** () const
- void **cacheNote** (const Note &note)
- void **expungeNote** (const Note &note)
- const Note * **findNote** (const QString &uid, const WhichUid whichUid) const
- void **clearAllNotes** ()
- size_t **numCachedResources** () const
- void **cacheResource** (const Resource &resource)
- void **expungeResource** (const Resource &resource)
- const Resource * **findResource** (const QString &id, const WhichUid whichUid) const
- void **clearAllResources** ()
- size_t **numCachedNotebooks** () const
- void **cacheNotebook** (const Notebook &notebook)
- void **expungeNotebook** (const Notebook &notebook)
- const Notebook * **findNotebook** (const QString &uid, const WhichUid whichUid) const
- const Notebook * **findNotebookByName** (const QString &name) const
- void **clearAllNotebooks** ()
- size_t **numCachedTags** () const
- void **cacheTag** (const Tag &tag)
- void **expungeTag** (const Tag &tag)
- const Tag * **findTag** (const QString &uid, const WhichUid whichUid) const
- const Tag * **findTagByName** (const QString &name) const
- void **clearAllTags** ()
- size_t **numCachedLinkedNotebooks** () const
- void **cacheLinkedNotebook** (const LinkedNotebook &linkedNotebook)
- void **expungeLinkedNotebook** (const LinkedNotebook &linkedNotebook)
- const LinkedNotebook * **findLinkedNotebook** (const QString &guid) const
- void **clearAllLinkedNotebooks** ()
- size_t **numCachedSavedSearches** () const
- void **cacheSavedSearch** (const SavedSearch &savedSearch)
- void **expungeSavedSearch** (const SavedSearch &savedSearch)
- const SavedSearch * **findSavedSearch** (const QString &uid, const WhichUid whichUid) const
- const SavedSearch * **findSavedSearchByName** (const QString &name) const
- void **clearAllSavedSearches** ()
- void **installCacheExpiryFunction** (const ILocalStorageCacheExpiryChecker &checker)
- virtual QTextStream & **print** (QTextStream &strm) const override

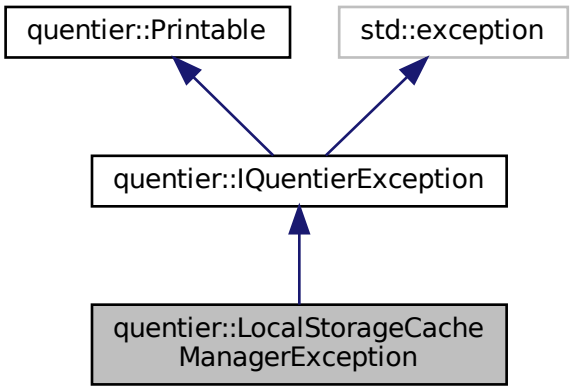
Additional Inherited Members

4.41 quentier::LocalStorageCacheManagerException Class Reference

Inheritance diagram for quentier::LocalStorageCacheManagerException:



Collaboration diagram for quentier::LocalStorageCacheManagerException:



Public Member Functions

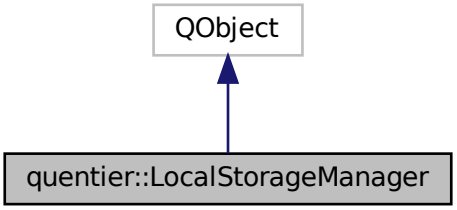
- **LocalStorageCacheManagerException** (const ErrorString &message)

Protected Member Functions

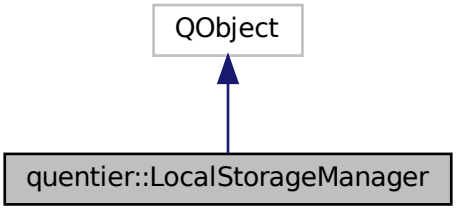
- virtual const QString **exceptionDisplayName** () const override

4.42 quentier::LocalStorageManager Class Reference

Inheritance diagram for quentier::LocalStorageManager:



Collaboration diagram for quentier::LocalStorageManager:



Public Types

- enum StartupOption { StartupOption::ClearDatabase = 1, StartupOption::OverrideLock = 2 }

  The StartupOption enum is a QFlags enum which allows to specify some options to be applied to the local storage database on startup or on call to switchUser method.
- enum ListObjectsOption {
  ListAll = 0, **ListDirty** = 1, **ListNonDirty** = 2, **ListElementsWithoutGuid** = 4,
  ListElementsWithGuid = 8, **ListLocal** = 16, **ListNonLocal** = 32, **ListFavoritedElements** = 64,
  ListNonFavoritedElements = 128 }

  The ListObjectsOption enum is a QFlags enum which allows to specify the desired local storage elements in calls to methods listing them from the database.
- enum OrderDirection { **Ascending** = 0, **Descending** }

The OrderDirection enum specifies the direction of ordering of the results for methods listing the objects from the local storage database.

- enum ListNotebooksOrder { **ByUpdateSequenceNumber** = 0, **ByNotebookName**, **ByCreationTimestamp**, **ByModificationTimestamp**, **NoOrder** }

  The ListNotebooksOrder allows to specify the results ordering for methods listing notebooks from the local storage database.
- enum ListLinkedNotebooksOrder { **ByUpdateSequenceNumber** = 0, **ByShareName**, **ByUsername**, **NoOrder** }

  The ListLinkedNotebooksOrder enum allows to specify the results ordering for methods listing linked notebooks from local storage.
- enum NoteCountOption { **IncludeNonDeletedNotes** = 1, **IncludeDeletedNotes** = 2 }

  The NoteCountOption enum is a QFlags enum which allows to specify some options for methods returning note counts from local storage.
- enum UpdateNoteOption { UpdateNoteOption::UpdateResourceMetadata = 1, UpdateNoteOption::UpdateResourceBinaryData = 2, UpdateNoteOption::UpdateTags = 4 }

  The UpdateNoteOption enum is a QFlags enum which allows to specify which note fields should be updated when updateNote method is called.
- enum GetNoteOption { GetNoteOption::WithResourceMetadata = 1, GetNoteOption::WithResourceBinaryData = 2 }

  The GetNoteOption enum is a QFlags enum which allows to specify which note fields should be included when findNote or one of listNote∗ methods is called.
- enum ListNotesOrder { **ByUpdateSequenceNumber** = 0, **ByTitle**, **ByCreationTimestamp**, **ByModificationTimestamp**, **ByDeletionTimestamp**, **ByAuthor**, **BySource**, **BySourceApplication**, **ByReminderTime**, **ByPlaceName**, **NoOrder** }

  The ListNotesOrder enum allows to specify the results ordering for methods listing notes from the local storage database.
- enum ListTagsOrder { **ByUpdateSequenceNumber**, **ByName**, **NoOrder** }

  The ListTagsOrder enum allows to specify the results ordering for methods listing tags from the local storage database.
- enum GetResourceOption { GetResourceOption::WithBinaryData = 1 }

  The GetResourceOption enum is a QFlags enum which allows to specify which resource fields should be included when findEnResource method is called.
- enum ListSavedSearchesOrder { **ByUpdateSequenceNumber** = 0, **ByName**, **ByFormat**, **NoOrder** }

  The ListSavedSearchesOrder enum allows to specify the results ordering for methods listing saved searches from local storage.

Signals

- void upgradeProgress (double progress)

  LocalStorageManager is capable of performing automatic database upgrades if/when it is necessary.

Public Member Functions

- LocalStorageManager (const Account &account, const StartupOptions options={}, QObject ∗parent=nullptr)

  LocalStorageManager - constructor. Takes in the account for which the LocalStorageManager instance is created plus some other parameters determining the startup behaviour.
- void switchUser (const Account &account, const StartupOptions options={})

  switchUser - switches to another local storage database file associated with the passed in account
- bool isLocalStorageVersionTooHigh (ErrorString &errorDescription)
- bool localStorageRequiresUpgrade (ErrorString &errorDescription)
- QVector< std::shared_ptr< ILocalStoragePatch > > requiredLocalStoragePatches ()

- qint32 localStorageVersion (ErrorString &errorDescription)
- qint32 highestSupportedLocalStorageVersion () const
- int userCount (ErrorString &errorDescription) const

  userCount returns the number of non-deleted users currently stored in the local storage database
- bool addUser (const User &user, ErrorString &errorDescription)

  addUser adds the passed in User object to the local storage database
- bool updateUser (const User &user, ErrorString &errorDescription)

  updateUser updates the passed in User object in the local storage database
- bool findUser (User &user, ErrorString &errorDescription) const

  findUser attempts to find and fill the fields of the passed in User object which must have "id" field set as this value is used as the identifier of User objects in the local storage database
- bool deleteUser (const User &user, ErrorString &errorDescription)

  deleteUser marks the user as deleted in local storage
- bool expungeUser (const User &user, ErrorString &errorDescription)

  expungeUser permanently deletes the user from the local storage database
- int notebookCount (ErrorString &errorDescription) const

  notebookCount returns the number of notebooks currently stored in the local storage database
- bool addNotebook (Notebook &notebook, ErrorString &errorDescription)

  addNotebook adds the passed in Notebook to the local storage database
- bool updateNotebook (Notebook &notebook, ErrorString &errorDescription)

  updateNotebook updates the passed in Notebook in the local storage database
- bool findNotebook (Notebook &notebook, ErrorString &errorDescription) const

  findNotebook attempts to find and set all found fields of the passed in Notebook object
- bool findDefaultNotebook (Notebook &notebook, ErrorString &errorDescription) const

  findDefaultNotebook attempts to find the default notebook in the local storage database.
- bool findLastUsedNotebook (Notebook &notebook, ErrorString &errorDescription) const

  findLastUsedNotebook attempts to find the last used notebook in the local storage database.
- bool findDefaultOrLastUsedNotebook (Notebook &notebook, ErrorString &errorDescription) const

  findDefaultOrLastUsedNotebook attempts to find either the default or the last used notebook in the local storage database.
- QList< Notebook > listAllNotebooks (ErrorString &errorDescription, const size_t limit=0, const size_t offset=0, const ListNotebooksOrder order=ListNotebooksOrder::NoOrder, const OrderDirection orderDirection=OrderDirection::Ascending, const QString &linkedNotebookGuid=QString()) const

  listAllNotebooks attempts to list all notebooks within the current account from the local storage database.
- QList< Notebook > listNotebooks (const ListObjectsOptions flag, ErrorString &errorDescription, const size_t limit=0, const size_t offset=0, const ListNotebooksOrder order=ListNotebooksOrder::NoOrder, const OrderDirection orderDirection=OrderDirection::Ascending, const QString &linkedNotebookGuid=QString()) const

  listNotebooks attempts to list notebooks within the account according to the specified input flag
- QList< SharedNotebook > listAllSharedNotebooks (ErrorString &errorDescription) const

  listAllSharedNotebooks attempts to list all shared notebooks within the account.
- QList< SharedNotebook > listSharedNotebooksPerNotebookGuid (const QString &notebookGuid, ErrorString &errorDescription) const

  listSharedNotebooksPerNotebookGuid - attempts to list all shared notebooks per given notebook's remote guid (not local uid, it's important).
- bool expungeNotebook (Notebook &notebook, ErrorString &errorDescription)

  expungeNotebook permanently deletes the specified notebook from the local storage database.
- int linkedNotebookCount (ErrorString &errorDescription) const

  linkedNotebookCount returns the number of linked notebooks stored in the local storage database.
- bool addLinkedNotebook (const LinkedNotebook &linkedNotebook, ErrorString &errorDescription)

  addLinkedNotebook adds passed in LinkedNotebook to the local storage database; LinkedNotebook must have "remote" Evernote service's guid set. It is not possible to add a linked notebook in offline mode so it doesn't make sense for LinkedNotebook objects to not have guid.

- bool updateLinkedNotebook (const LinkedNotebook &linkedNotebook, ErrorString &errorDescription)

  updateLinkedNotebook updates passd in LinkedNotebook in the local storage database; LinkedNotebook must have "remote" Evernote service's guid set.

- bool findLinkedNotebook (LinkedNotebook &linkedNotebook, ErrorString &errorDescription) const

  findLinkedNotebook attempts to find and set all found fields for passed in by reference LinkedNotebook object. For LinkedNotebook local uid doesn't mean anything because it can only be considered valid if it has "remote" Evernote service's guid set. So this passed in LinkedNotebook object must have guid set to identify the linked notebook in the local storage database.

- QList< LinkedNotebook > listAllLinkedNotebooks (ErrorString &errorDescription, const size_t limit=0, const size_t offset=0, const ListLinkedNotebooksOrder order=ListLinkedNotebooksOrder::NoOrder, const OrderDirection orderDirection=OrderDirection::Ascending) const

  listAllLinkedNotebooks - attempts to list all linked notebooks within the account.

- QList< LinkedNotebook > listLinkedNotebooks (const ListObjectsOptions flag, ErrorString &errorDescription, const size_t limit=0, const size_t offset=0, const ListLinkedNotebooksOrder order=ListLinkedNotebooksOrder::NoOrder, const OrderDirection orderDirection=OrderDirection::Ascending) const

  listLinkedNotebooks attempts to list linked notebooks within the account according to the specified input flag.

- bool expungeLinkedNotebook (const LinkedNotebook &linkedNotebook, ErrorString &errorDescription)

  expungeLinkedNotebook permanently deletes specified linked notebook from the local storage database.

- int noteCount (ErrorString &errorDescription, const NoteCountOptions options=NoteCountOption::IncludeNonDeletedNotes) const

  noteCount returns the number of notes currently stored in the local storage database.

- int noteCountPerNotebook (const Notebook &notebook, ErrorString &errorDescription, const NoteCountOptions options=NoteCountOption::IncludeNonDeletedNotes) const

  noteCountPerNotebook returns the number of notes currently stored in the local storage database per given notebook.

- int noteCountPerTag (const Tag &tag, ErrorString &errorDescription, const NoteCountOptions options=NoteCountOption::IncludeNonDeletedNotes) const

  noteCountPerTag returns the number of notes currently stored in local storage database labeled with given tag.

- bool noteCountsPerAllTags (QHash< QString, int > &noteCountsPerTagLocalUid, ErrorString &errorDescription, const NoteCountOptions options=NoteCountOption::IncludeNonDeletedNotes) const

  noteCountsPerAllTags returns the number of notes currently stored in local storage database labeled with each tag stored in the local storage database.

- int noteCountPerNotebooksAndTags (const QStringList &notebookLocalUids, const QStringList &tagLocalUids, ErrorString &errorDescription, const NoteCountOptions options=NoteCountOption::IncludeNonDeletedNotes) const

  noteCountPerNotebooksAndTags returns the number of notes currently stored in local storage database belonging to one of notebooks corresponding to given notebook local uids and labeled by at least one of tags corresponding to given tag local uids

- bool addNote (Note &note, ErrorString &errorDescription)

  addNote adds passed in Note to the local storage database.

- bool updateNote (Note &note, const UpdateNoteOptions options, ErrorString &errorDescription)

  updateNote updates passed in Note in the local storage database.

- bool findNote (Note &note, const GetNoteOptions options, ErrorString &errorDescription) const

  findNote - attempts to find note in the local storage database

- QList< Note > listNotesPerNotebook (const Notebook &notebook, const GetNoteOptions options, ErrorString &errorDescription, const ListObjectsOptions &flag=ListObjectsOption::ListAll, const size_t limit=0, const size_t offset=0, const ListNotesOrder &order=ListNotesOrder::NoOrder, const OrderDirection &orderDirection=OrderDirection::Ascending) const

  listNotesPerNotebook attempts to list notes per given notebook

- QList< Note > listNotesPerTag (const Tag &tag, const GetNoteOptions options, ErrorString &errorDescription, const ListObjectsOptions &flag=ListObjectsOption::ListAll, const size_t limit=0, const size_t offset=0, const ListNotesOrder &order=ListNotesOrder::NoOrder, const OrderDirection &orderDirection=OrderDirection::Ascending) const

  listNotesPerTag attempts to list notes labeled with a given tag

- QList< Note > listNotesPerNotebooksAndTags (const QStringList &notebookLocalUids, const QString↩ List &tagLocalUids, const GetNoteOptions options, ErrorString &errorDescription, const ListObjects↩ Options &flag=ListObjectsOption::ListAll, const size_t limit=0, const size_t offset=0, const ListNotesOrder &order=ListNotesOrder::NoOrder, const OrderDirection &orderDirection=OrderDirection::Ascending) const

  listNotesPerNotebooksAndTags attempts to list notes which are present within one of specified notebooks and are labeled with at least one of specified tags
- QList< Note > listNotesByLocalUids (const QStringList &noteLocalUids, const GetNoteOptions options, ErrorString &errorDescription, const ListObjectsOptions &flag=ListObjectsOption::ListAll, const size_t limit=0, const size_t offset=0, const ListNotesOrder &order=ListNotesOrder::NoOrder, const OrderDirection &order↩ Direction=OrderDirection::Ascending) const

  listNotesByLocalUids attempts to list notes given their local uids
- QList< Note > listNotes (const ListObjectsOptions flag, const GetNoteOptions options, ErrorString &error↩ Description, const size_t limit=0, const size_t offset=0, const ListNotesOrder order=ListNotesOrder::NoOrder, const OrderDirection orderDirection=OrderDirection::Ascending, const QString &linkedNotebookGuid=Q↩ String()) const

  listNotes attempts to list notes within the account according to the specified input flag.
- QStringList findNoteLocalUidsWithSearchQuery (const NoteSearchQuery &noteSearchQuery, ErrorString &errorDescription) const

  findNoteLocalUidsWithSearchQuery attempts to find note local uids of notes corresponding to the passed in Note↩ SearchQuery object.
- NoteList findNotesWithSearchQuery (const NoteSearchQuery &noteSearchQuery, const GetNoteOptions options, ErrorString &errorDescription) const

  findNotesWithSearchQuery attempts to find notes corresponding to the passed in NoteSearchQuery object.
- bool expungeNote (Note &note, ErrorString &errorDescription)

  expungeNote permanently deletes note from local storage.
- int tagCount (ErrorString &errorDescription) const

  tagCount returns the number of non-deleted tags currently stored in the local storage database.
- bool addTag (Tag &tag, ErrorString &errorDescription)

  addTag adds passed in Tag to the local storage database. If tag has "remote" Evernote service's guid set, it is identified in the database by this guid. Otherwise it is identified by local uid.
- bool updateTag (Tag &tag, ErrorString &errorDescription)

  updateTag updates passed in Tag in the local storage database.
- bool findTag (Tag &tag, ErrorString &errorDescription) const

  findTag attempts to find and fill the fields of passed in tag object.
- QList< Tag > listAllTagsPerNote (const Note &note, ErrorString &errorDescription, const ListObjects↩ Options &flag=ListObjectsOption::ListAll, const size_t limit=0, const size_t offset=0, const ListTagsOrder &order=ListTagsOrder::NoOrder, const OrderDirection &orderDirection=OrderDirection::Ascending) const

  listAllTagsPerNote lists all tags per given note
- QList< Tag > listAllTags (ErrorString &errorDescription, const size_t limit=0, const size_t offset=0, const ListTagsOrder order=ListTagsOrder::NoOrder, const OrderDirection orderDirection=OrderDirection::↩ Ascending, const QString &linkedNotebookGuid=QString()) const

  listAllTags lists all tags within the current user's account.
- QList< Tag > listTags (const ListObjectsOptions flag, ErrorString &errorDescription, const size_t limit=0, const size_t offset=0, const ListTagsOrder &order=ListTagsOrder::NoOrder, const OrderDirection order↩ Direction=OrderDirection::Ascending, const QString &linkedNotebookGuid=QString()) const

  listTags attempts to list tags within the account according to the specified input flag.
- QList< std::pair< Tag, QStringList > > listTagsWithNoteLocalUids (const ListObjectsOptions flag, Error↩ String &errorDescription, const size_t limit=0, const size_t offset=0, const ListTagsOrder &order=ListTags↩ Order::NoOrder, const OrderDirection orderDirection=OrderDirection::Ascending, const QString &linked↩ NotebookGuid=QString()) const

  listTagsWithNoteLocalUids attempts to list tags and their corresponding local uids within the account according to the specified input flag
- bool expungeTag (Tag &tag, QStringList &expungedChildTagLocalUids, ErrorString &errorDescription)

  expungeTag permanently deletes tag from the local storage database.

- bool expungeNotelessTagsFromLinkedNotebooks (ErrorString &errorDescription)

  expungeNotelessTagsFromLinkedNotebooks permanently deletes from the local storage database those tags which belong to some linked notebook and are not linked with any notes.
- int enResourceCount (ErrorString &errorDescription) const

  enResourceCount (the name is not Resource to prevent problems with macro defined on some versions of Windows) returns the number of resources currently stored in the local storage database.
- bool addEnResource (Resource &resource, ErrorString &errorDescription)

  addEnResource adds passed in resource to the local storage database.
- bool updateEnResource (Resource &resource, ErrorString &errorDescription)

  updateEnResource updates passed in resource in the local storage database.
- bool findEnResource (Resource &resource, const GetResourceOptions options, ErrorString &errorDescription) const

  findEnResource method attempts to find resource in the local storage database
- bool expungeEnResource (Resource &resource, ErrorString &errorDescription)

  expungeResource permanently deletes resource from the local storage database.
- int savedSearchCount (ErrorString &errorDescription) const

  savedSearchCount returns the number of saved seacrhes currently stored in local storage database.
- bool addSavedSearch (SavedSearch &search, ErrorString &errorDescription)

  addSavedSearch adds passed in SavedSearch to the local storage database; if search has "remote" Evernote service's guid set, it is identified in the database by this guid. Otherwise it is identified by local uid.
- bool updateSavedSearch (SavedSearch &search, ErrorString &errorDescription)

  updateSavedSearch updates passed in SavedSearch in the local storage database.
- bool findSavedSearch (SavedSearch &search, ErrorString &errorDescription) const

  findSavedSearch attempts to find and fill the fields of passed in saved search object.
- QList< SavedSearch > listAllSavedSearches (ErrorString &errorDescription, const size_t limit=0, const size_t offset=0, const ListSavedSearchesOrder order=ListSavedSearchesOrder::NoOrder, const OrderDirection orderDirection=OrderDirection::Ascending) const

  listAllSavedSearches lists all saved searches within the account.
- QList< SavedSearch > listSavedSearches (const ListObjectsOptions flag, ErrorString &errorDescription, const size_t limit=0, const size_t offset=0, const ListSavedSearchesOrder order=ListSavedSearchesOrder::NoOrder, const OrderDirection orderDirection=OrderDirection::Ascending) const

  listSavedSearches attempts to list saved searches within the account according to the specified input flag.
- bool expungeSavedSearch (SavedSearch &search, ErrorString &errorDescription)

  expungeSavedSearch permanently deletes saved search from the local storage database.
- qint32 accountHighUsn (const QString &linkedNotebookGuid, ErrorString &errorDescription)

  accountHighUsn returns the highest update sequence number within the data elements stored in the local storage database, either for user's own account or for some linked notebook.

Friends

- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const StartupOption option)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &dbg, const StartupOption option)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const StartupOptions options)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &dbg, const StartupOptions options)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const ListObjectsOption option)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &dbg, const ListObjectsOption option)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const ListObjectsOptions options)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &dbg, const ListObjectsOptions options)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const OrderDirection orderDirection)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &dbg, const OrderDirection orderDirection)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const ListNotebooksOrder order)

- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &dbg, const ListNotebooksOrder order)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const ListLinkedNotebooksOrder order)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &strm, const ListLinkedNotebooksOrder order)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const NoteCountOption option)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &dbg, const NoteCountOption option)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const NoteCountOptions options)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &strm, const NoteCountOptions options)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const UpdateNoteOption option)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &strm, const UpdateNoteOption option)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const UpdateNoteOptions options)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &strm, const UpdateNoteOptions options)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const GetNoteOption option)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &dbg, const GetNoteOption option)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const GetNoteOptions options)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &strm, const GetNoteOptions options)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const ListNotesOrder order)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &strm, const ListNotesOrder order)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const ListTagsOrder order)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &strm, const ListTagsOrder order)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const GetResourceOption option)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &strm, const GetResourceOption option)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const GetResourceOptions options)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &strm, const GetResourceOptions options)
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const ListSavedSearchesOrder order)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &strm, const ListSavedSearchesOrder order)

4.42.1 Member Enumeration Documentation

4.42.1.1 GetNoteOption

`enum quentier::LocalStorageManager::GetNoteOption [strong]`

The GetNoteOption enum is a QFlags enum which allows to specify which note fields should be included when findNote or one of listNote∗ methods is called.

Most note data is included unconditionally - note title, content, attributes (if any) etc. However, some specific data can be opted to not be included into the returned note data - notably, metadata of resources and binary data of resources. If these are omitted, findNote or any of listNote∗ methods might work faster than otherwise

Enumerator

WithResourceMetadata	WithResourceMetadata value specifies that fields aside dataBody, dataSize, dataHash, alternateDataBody, alternateDataSize, alternateDataHash for each note's resource should be included
WithResourceBinaryData	WithResourceBinaryData value specifies that dataBody, its size and hash and alternateDataBody, its size and hash should be included into each of note's resources; this value only has effect if flags also have WithResourceMetadata value enabled!

4.42.1.2 GetResourceOption

enum quentier::LocalStorageManager::GetResourceOption [strong]

The GetResourceOption enum is a QFlags enum which allows to specify which resource fields should be included when findEnResource method is called.

Most resource data is included unconditionally but some specific data can be opted to not be included into the returned resource data - notably, binary data of the resource. If it is omitted, findEnResource method might work faster than otherwise

Enumerator

WithBinaryData	WithBinaryData value specifies than dataBody and alternateDataBody should be included into the returned resource

4.42.1.3 ListObjectsOption

enum quentier::LocalStorageManager::ListObjectsOption [strong]

The ListObjectsOption enum is a QFlags enum which allows to specify the desired local storage elements in calls to methods listing them from the database.

For example, one can either list all available elements of certain type from local storage or only elements marked as dirty (modified locally, not yet synchronized) or elements never synchronized with the remote storage or elements which are synchronizable with the remote storage etc.

4.42.1.4 StartupOption

enum quentier::LocalStorageManager::StartupOption [strong]

The StartupOption enum is a QFlags enum which allows to specify some options to be applied to the local storage database on startup or on call to switchUser method.

Enumerator

ClearDatabase	If ClearDatabase flag is active, LocalStorageManager would wipe any existing database contents; the net effect would be as if no database existed for the given user before the creation of LocalStorageManager or before the call to its switchUser method
OverrideLock	If OverrideLock flag is active, LocalStorageManager would ignore the existing advisory lock (if any) put on the database file; if this flag is not active, the attempt to create LocalStorageManager (or the attempt to call its switchUser method) with the advisory lock on the database file put by someone else would cause the throwing of DatabaseLockedException

4.42.1.5 UpdateNoteOption

```
enum quentier::LocalStorageManager::UpdateNoteOption  [strong]
```

The UpdateNoteOption enum is a QFlags enum which allows to specify which note fields should be updated when updateNote method is called.

Most note data is updated unconditionally - note title, content, attributes (if any) etc. However, some specific data can be chosen to not update - notably, metadata of resources, binary data of resources or lists of note's tags

Enumerator

UpdateResourceMetadata	UpdateResourceMetadata value specifies that fields aside dataBody, dataSize, dataHash, alternateDataBody, alternateDataSize, alternateDataHash for each note's resource should be updated
UpdateResourceBinaryData	UpdateResourceBinaryData value specifies that dataBody, its size and hash and alternateDataBody, its size and hash should be updated for each of note's resources; this value only has effect if flags also have UpdateResourceMetadata value enabled!
UpdateTags	UpdateTags value specifies that note's tag lists should be updated

4.42.2 Constructor & Destructor Documentation

4.42.2.1 LocalStorageManager()

```
quentier::LocalStorageManager::LocalStorageManager (
            const Account & account,
            const StartupOptions options = {},
            QObject * parent = nullptr )  [explicit]
```

LocalStorageManager - constructor. Takes in the account for which the LocalStorageManager instance is created plus some other parameters determining the startup behaviour.

Parameters

account	The account for which the local storage is being created and initialized
options	Startup options for the local storage, none enabled by default
parent	Parent QObject

4.42.3 Member Function Documentation

4.42.3.1 accountHighUsn()

```
qint32 quentier::LocalStorageManager::accountHighUsn (
            const QString & linkedNotebookGuid,
            ErrorString & errorDescription )
```

accountHighUsn returns the highest update sequence number within the data elements stored in the local storage database, either for user's own account or for some linked notebook.

Parameters

linkedNotebookGuid	The guid of the linked notebook for which the highest update sequence number is requested; if null or empty, the highest update sequence number for user's own account is returned
errorDescription	Error description if account's highest update sequence number could not be returned

Returns

Either the highest update sequence number - a non-negative value - or a negative number in case of error

4.42.3.2 addEnResource()

```
bool quentier::LocalStorageManager::addEnResource (
            Resource & resource,
            ErrorString & errorDescription )
```

addEnResource adds passed in resource to the local storage database.

Parameters

resource	Resource to be added to the database, must have either note's local uid set or note's "remote" Evernote service's guid set; may be changed as a result of the call, filled with autogenerated fields like local uid if it was empty before the call
errorDescription	Error description if resource could not be added

Returns

True if resource was added successfully, false otherwise

4.42.3.3 addLinkedNotebook()

```
bool quentier::LocalStorageManager::addLinkedNotebook (
            const LinkedNotebook & linkedNotebook,
            ErrorString & errorDescription )
```

addLinkedNotebook adds passed in LinkedNotebook to the local storage database; LinkedNotebook must have "remote" Evernote service's guid set. It is not possible to add a linked notebook in offline mode so it doesn't make sense for LinkedNotebook objects to not have guid.

Parameters

linkedNotebook	LinkedNotebook to be added to the local storage database
errorDescription	Error description if linked notebook could not be added

Returns

True if linked notebook was added successfully, false otherwise

4.42.3.4 addNote()

```
bool quentier::LocalStorageManager::addNote (
            Note & note,
            ErrorString & errorDescription )
```

addNote adds passed in Note to the local storage database.

Parameters

note	Note to be added to local storage database; required to contain either "remote" notebook guid or local notebook uid; may be changed as a result of the call, filled with autogenerated fields like local uid if it was empty before the call; also tag guids are filled if the note passed in contained only tag local uids and tag local uids are filled if the note passed in contained only tag guids
errorDescription	Error description if note could not be added

Returns

True if note was added successfully, false otherwise

4.42.3.5 addNotebook()

```
bool quentier::LocalStorageManager::addNotebook (
            Notebook & notebook,
            ErrorString & errorDescription )
```

addNotebook adds the passed in Notebook to the local storage database

If the notebook has "remote" Evernote service's guid set, it is identified by this guid in the local storage database. Otherwise it is identified by the local uid

Parameters

notebook	The notebook to be added to the local storage database; the object is passed by reference and may be changed as a result of the call (filled with autocompleted fields like local uid if it was empty before the call)
errorDescription	Error description if the notebook could not be added

Returns

True if the notebook was added successfully, false otherwise

4.42.3.6 addSavedSearch()

```
bool quentier::LocalStorageManager::addSavedSearch (
            SavedSearch & search,
            ErrorString & errorDescription )
```

addSavedSearch adds passed in SavedSearch to the local storage database; if search has "remote" Evernote service's guid set, it is identified in the database by this guid. Otherwise it is identified by local uid.

Parameters

search	SavedSearch to be added to the local storage; may be changed as a result of the call, filled with autogenerated fields like local uid if it was empty before the call
errorDescription	Error description if SavedSearch could not be added

Returns

True if SavedSearch was added successfully, false otherwise

4.42.3.7 addTag()

```
bool quentier::LocalStorageManager::addTag (
            Tag & tag,
            ErrorString & errorDescription )
```

addTag adds passed in Tag to the local storage database. If tag has "remote" Evernote service's guid set, it is identified in the database by this guid. Otherwise it is identified by local uid.

Parameters

tag	Tag to be added to the local storage; may be changed as a result of the call, filled with autogenerated fields like local uid if it was empty before the call
errorDescription	Error description if Tag could not be added

Returns

True if Tag was added successfully, false otherwise

4.42.3.8 addUser()

```
bool quentier::LocalStorageManager::addUser (
            const User & user,
            ErrorString & errorDescription )
```

addUser adds the passed in User object to the local storage database

The table with Users is only involved in operations with notebooks which have "contact" field set which in turn is used with business accounts

Parameters

user	The user to be added to the local storage database
errorDescription	Error description if the user could not be added

Returns

True if the user was added successfully, false otherwise

4.42.3.9 deleteUser()

```
bool quentier::LocalStorageManager::deleteUser (
            const User & user,
            ErrorString & errorDescription )
```

deleteUser marks the user as deleted in local storage

Parameters

user	The user to be marked as deleted
errorDescription	Error description if the user could not be marked as deleted

Returns

True if the user was marked as deleted successfully, false otherwise

4.42.3.10 enResourceCount()

```
int quentier::LocalStorageManager::enResourceCount (
            ErrorString & errorDescription ) const
```

enResourceCount (the name is not Resource to prevent problems with macro defined on some versions of Windows) returns the number of resources currently stored in the local storage database.

Parameters

errorDescription	Error description if the number of resources could not be returned

Returns

Either non-negative value with the number of resources or -1 which means some error occurred

4.42.3.11 expungeEnResource()

```
bool quentier::LocalStorageManager::expungeEnResource (
            Resource & resource,
            ErrorString & errorDescription )
```

expungeResource permanently deletes resource from the local storage database.

Parameters

resource	Resource to be expunged; may be changed as a result of the call, automatically filled with local uid and note local uid and/or guid if these were empty before the call
errorDescription	Error description if resource could not be expunged

Returns

True if resource was expunged successfully, false otherwise

4.42.3.12 expungeLinkedNotebook()

```
bool quentier::LocalStorageManager::expungeLinkedNotebook (
            const LinkedNotebook & linkedNotebook,
            ErrorString & errorDescription )
```

expungeLinkedNotebook permanently deletes specified linked notebook from the local storage database.

Evernote API doesn't allow to delete linked notebooks from the remote storage, it can only be done by official desktop client or web client. So this method should be called only during the synchronization with remote service, when some linked notebook is found to be deleted via either official desktop client or web cient.

Parameters

linkedNotebook	Linked notebook to be expunged. Must have "remote" guid set
errorDescription	Error description if linked notebook could not be expunged

Returns

True if linked notebook was expunged successfully, false otherwise

4.42.3.13 expungeNote()

```
bool quentier::LocalStorageManager::expungeNote (
            Note & note,
            ErrorString & errorDescription )
```

expungeNote permanently deletes note from local storage.

Evernote API doesn't allow to delete notes from the remote storage, it can only be done by official desktop client or web client. So this method should be called only during the synchronization with remote database, when some note is found to be deleted via either official desktop client or web client.

Parameters

note	Note to be expunged; may be changed as a result of the call, filled with fields like local uid or notebook guid or local uid
errorDescription	Error description if note could not be expunged

Returns

True if note was expunged successfully, false otherwise

4.42.3.14 expungeNotebook()

```
bool quentier::LocalStorageManager::expungeNotebook (
            Notebook & notebook,
            ErrorString & errorDescription )
```

expungeNotebook permanently deletes the specified notebook from the local storage database.

Evernote API doesn't allow to delete the notebooks from the remote storage, it can only be done by the official desktop Evernote client or via its web client. So this method should be called only during the synchronization with the remote storage, when some notebook is found to be deleted via either the official desktop client or via the web client; also, this method can be called for local notebooks not synchronized with Evernote at all.

Parameters

notebook	The notebook to be expunged. Must have either "remote" guid or local uid set; the object is passed by reference and may be changed as a result of the call (filled with local uid if it was empty before the call)
errorDescription	Error description if the notebook could not be expunged

Returns

True if the notebook was expunged successfully, false otherwise

4.42.3.15 expungeNotelessTagsFromLinkedNotebooks()

```
bool quentier::LocalStorageManager::expungeNotelessTagsFromLinkedNotebooks (
            ErrorString & errorDescription )
```

expungeNotelessTagsFromLinkedNotebooks permanently deletes from the local storage database those tags which belong to some linked notebook and are not linked with any notes.

Parameters

errorDescription	Error description if tag could not be expunged

Returns

True if relevant tags were expunged successfully, false otherwise

4.42.3.16 expungeSavedSearch()

```
bool quentier::LocalStorageManager::expungeSavedSearch (
            SavedSearch & search,
            ErrorString & errorDescription )
```

expungeSavedSearch permanently deletes saved search from the local storage database.

Parameters

search	Saved search to be expunged; may be changed as a result of the call filled local uid if it was empty before the call
errorDescription	Error description if saved search could not be expunged

Returns

True if saved search was expunged successfully, false otherwise

4.42.3.17 expungeTag()

```
bool quentier::LocalStorageManager::expungeTag (
            Tag & tag,
```

```
            QStringList & expungedChildTagLocalUids,
            ErrorString & errorDescription )
```

expungeTag permanently deletes tag from the local storage database.

Evernote API doesn't allow to delete tags from remote storage, it can only be done by official desktop client or web client. So this method should be called only during the synchronization with remote database, when some tag is found to be deleted via either official desktop client or web client.

Parameters

tag	Tag to be expunged; may be changed as a result of the call, automatically filled with local uid if it was empty before the call
expungedChildTagLocalUids	If the expunged tag was a parent of some other tags, these were expunged as well; this parameter would contain the local uids of expunged child tags
errorDescription	Error description if tag could not be expunged

Returns

True if tag was expunged successfully, false otherwise

4.42.3.18 expungeUser()

```
bool quentier::LocalStorageManager::expungeUser (
            const User & user,
            ErrorString & errorDescription )
```

expungeUser permanently deletes the user from the local storage database

Parameters

user	The user to be expunged
errorDescription	Error description if the user could not be expunged

Returns

True if the user was expunged successfully, false otherwise

4.42.3.19 findDefaultNotebook()

```
bool quentier::LocalStorageManager::findDefaultNotebook (
            Notebook & notebook,
            ErrorString & errorDescription ) const
```

findDefaultNotebook attempts to find the default notebook in the local storage database.

Parameters

notebook	The default notebook to be found
errorDescription	Error description if the default notebook could not be found

Returns

True if the default notebook was found, false otherwise

4.42.3.20 findDefaultOrLastUsedNotebook()

```
bool quentier::LocalStorageManager::findDefaultOrLastUsedNotebook (
            Notebook & notebook,
            ErrorString & errorDescription ) const
```

findDefaultOrLastUsedNotebook attempts to find either the default or the last used notebook in the local storage database.

Parameters

notebook	Either the default or the last used notebook to be found
errorDescription	Error description if the default or the last used notebook could not be found

Returns

True if the default or the last used notebook were found, false otherwise

4.42.3.21 findEnResource()

```
bool quentier::LocalStorageManager::findEnResource (
            Resource & resource,
            const GetResourceOptions options,
            ErrorString & errorDescription ) const
```

findEnResource method attempts to find resource in the local storage database

Parameters

resource	Resource to be found in the local storage database. If it has the "remote" Evernote service's guid set, this guid is used to identify the resource in the local storage database. Otherwise resource's local uid is used
options	Options specifying which optionally includable fields of the resource should actually be included
errorDescription	Error description if resource could not be found

Returns

True if resource was found successfully, false otherwise

4.42.3.22 findLastUsedNotebook()

```
bool quentier::LocalStorageManager::findLastUsedNotebook (
            Notebook & notebook,
            ErrorString & errorDescription ) const
```

findLastUsedNotebook attempts to find the last used notebook in the local storage database.

Parameters

notebook	The last used notebook to be found
errorDescription	Error description if the last used notebook could not be found

Returns

True if the last used notebook was found, false otherwise

4.42.3.23 findLinkedNotebook()

```
bool quentier::LocalStorageManager::findLinkedNotebook (
            LinkedNotebook & linkedNotebook,
            ErrorString & errorDescription ) const
```

findLinkedNotebook attempts to find and set all found fields for passed in by reference LinkedNotebook object. For LinkedNotebook local uid doesn't mean anything because it can only be considered valid if it has "remote" Evernote service's guid set. So this passed in LinkedNotebook object must have guid set to identify the linked notebook in the local storage database.

Parameters

linkedNotebook	Linked notebook to be found. Must have "remote" guid set
errorDescription	Error description if linked notebook could not be found

Returns

True if linked notebook was found, false otherwise

4.42.3.24 findNote()

```
bool quentier::LocalStorageManager::findNote (
            Note & note,
            const GetNoteOptions options,
            ErrorString & errorDescription ) const
```

findNote - attempts to find note in the local storage database

Parameters

note	- note to be found in the local storage database. Must have either local or "remote" Evernote service's guid set
options	- options specifying which optionally includable fields of the note should actually be included
errorDescription	- error description if note could not be found

Returns

true if note was found successfully, false otherwise

4.42.3.25 findNotebook()

```
bool quentier::LocalStorageManager::findNotebook (
            Notebook & notebook,
            ErrorString & errorDescription ) const
```

findNotebook attempts to find and set all found fields of the passed in Notebook object

If "remote" Evernote service's guid for the notebook is set, it is used to identify the notebook in the local storage database. Otherwise the notebook is identified by its local uid. If it's empty, the search would attempt to find the notebook by its name. If the name is also not set, the search would attempt to find the notebook by linked notebook guid assuming that no more than one notebook corresponds to the linked notebook guid. If linked notebook guid is also not set, the search would fail.

Important! Due to the fact that the notebook name is only unique within the users's own account as well as within each linked notebook, the result of the search by name depends on the notebook's linked notebook guid: if it is not set, the search by name would only search for the notebook with the specified name within the user's own account. If it is set, the search would only consider the linked notebook with the corresponding guid.

Parameters

notebook	The notebook to be found. Must have either "remote" or local uid or name or linked notebook guid set
errorDescription	Error description if the notebook could not be found

Returns

True if the notebook was found, false otherwise

4.42.3.26 findNoteLocalUidsWithSearchQuery()

```
QStringList quentier::LocalStorageManager::findNoteLocalUidsWithSearchQuery (
            const NoteSearchQuery & noteSearchQuery,
            ErrorString & errorDescription ) const
```

findNoteLocalUidsWithSearchQuery attempts to find note local uids of notes corresponding to the passed in NoteSearchQuery object.

Parameters

noteSearchQuery	Filled NoteSearchQuery object used to filter the notes
errorDescription	Error description in case note local uids could not be listed

Returns

The list of found notes' local uids or empty list in case of error

4.42.3.27 findNotesWithSearchQuery()

```
NoteList quentier::LocalStorageManager::findNotesWithSearchQuery (
            const NoteSearchQuery & noteSearchQuery,
            const GetNoteOptions options,
            ErrorString & errorDescription ) const
```

findNotesWithSearchQuery attempts to find notes corresponding to the passed in NoteSearchQuery object.

Parameters

noteSearchQuery	Filled NoteSearchQuery object used to filter the notes
options	Options specifying which optionally includable fields of the note should actually be included
errorDescription	Error description in case notes could not be listed

Returns

Either list of notes per NoteSearchQuery or empty list in case of error or no notes presence for the given NoteSearchQuery

4.42.3.28 findSavedSearch()

```
bool quentier::LocalStorageManager::findSavedSearch (
            SavedSearch & search,
            ErrorString & errorDescription ) const
```

findSavedSearch attempts to find and fill the fields of passed in saved search object.

If "remote" Evernote services's guid for the saved search is set, it would be used to identify the saved search in the local storage. Otherwise the local uid would be used. If neither guid not local uid are set, saved search's name would be used. If the name is also not set, the search for saved search would fail.

Parameters

search	SavedSearch to be found in the local storage database
errorDescription	Error description if SavedSearch could not be found

Returns

True if SavedSearch was found, false otherwise

4.42.3.29 findTag()

```
bool quentier::LocalStorageManager::findTag (
            Tag & tag,
            ErrorString & errorDescription ) const
```

findTag attempts to find and fill the fields of passed in tag object.

If "remote" Evernote service's guid for the tag is set, it would be used to identify the tag in the local storage database. Otherwise the local uid would be used. If neither guid nor local uid are set, tag's name would be used. If the name is also not set, the search would fail.

Important! Due to the fact that the tag name is only unique within the users's own account as well as within each linked notebook, the result of the search by name depends on the tag's linked notebook guid: if it is not set, the search by name would only search for the tag with the specified name within the user's own account. If it is set, the search would only consider tags from a linked notebook with the corresponding guid.

Parameters

tag	Tag to be found in the local storage database; must have either guid, local uid or name set
errorDescription	Error description in case tag could not be found

Returns

True if tag was found, false otherwise

4.42.3.30 findUser()

```
bool quentier::LocalStorageManager::findUser (
            User & user,
            ErrorString & errorDescription ) const
```

findUser attempts to find and fill the fields of the passed in User object which must have "id" field set as this value is used as the identifier of User objects in the local storage database

Parameters

user	The user to be found. Must have "id" field set
errorDescription	Error description if the user could not be found

Returns

True if the user was found successfully, false otherwise

4.42.3.31 highestSupportedLocalStorageVersion()

```
qint32 quentier::LocalStorageManager::highestSupportedLocalStorageVersion ( ) const
```

highestSupportedLocalStorageVersion returns the highest version of local storage persistence which the current build of libquentier is capable of working with

Returns

Highest supported local storage version

4.42.3.32 isLocalStorageVersionTooHigh()

```
bool quentier::LocalStorageManager::isLocalStorageVersionTooHigh (
            ErrorString & errorDescription )
```

isLocalStorageVersionTooHigh method checks whether the existing local storage persistence has version which is too high for the currenly run version of libquentier to work with i.e. whether the local storage has already been upgraded using a new version of libquentier.

NOTE: it is libquentier client code's responsibility to call this method and/or localStorageRequiresUpgrade method, libquentier won't call any of these on its own and will just attempt to work with the existing local storage, whatever version it is of. If version is too high, things can fail in most mysterious way, so the client code is obliged to call these methods to ensure the local storage version is checked properly.

Parameters

errorDescription	Textual description of the error if the method was unable to determine whether the local storage version is too high for the currently run version of libquentier to work with, otherwise this parameter is not touched by the method

Returns

True if local storage version is too high for the currently run version of libquentier to work with, false otherwise

4.42.3.33 linkedNotebookCount()

```
int quentier::LocalStorageManager::linkedNotebookCount (
            ErrorString & errorDescription ) const
```

linkedNotebookCount returns the number of linked notebooks stored in the local storage database.

Parameters

errorDescription	Error description if the number of linked notebooks count not be returned

Returns

Either non-negative number of linked notebooks or -1 if some error has occurred

4.42.3.34 listAllLinkedNotebooks()

```
QList<LinkedNotebook> quentier::LocalStorageManager::listAllLinkedNotebooks (
            ErrorString & errorDescription,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListLinkedNotebooksOrder order = ListLinkedNotebooksOrder::NoOrder,
            const OrderDirection orderDirection = OrderDirection::Ascending ) const
```

listAllLinkedNotebooks - attempts to list all linked notebooks within the account.

Parameters

errorDescription	Error description if linked notebooks could not be listed, otherwise this parameter is untouched
limit	Limit for the max number of linked notebooks in the result, zero by default which means no limit is set
offset	Number of linked notebooks to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of linked notebooks in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder

Returns

Either list of all linked notebooks or empty list in case of error or no linked notebooks presence within the account

4.42.3.35 listAllNotebooks()

```
QList<Notebook> quentier::LocalStorageManager::listAllNotebooks (
            ErrorString & errorDescription,
```

```
            const size_t limit = 0,
            const size_t offset = 0,
            const ListNotebooksOrder order = ListNotebooksOrder::NoOrder,
            const OrderDirection orderDirection = OrderDirection::Ascending,
            const QString & linkedNotebookGuid = QString() ) const
```

listAllNotebooks attempts to list all notebooks within the current account from the local storage database.

Parameters

errorDescription	Error description if all notebooks could not be listed; if no error happens, this parameter is untouched
limit	The limit for the max number of notebooks in the result, zero by default which means no limit is set
offset	The number of notebooks to skip in the beginning of the result, zero by default
order	Allows to specify a particular ordering of notebooks in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder
linkedNotebookGuid	If it's null, the method would list the notebooks ignoring their belonging to the current account or to some linked notebook; if it's empty, only the non-linked notebooks would be listed; otherwise, the only one notebook from the corresponding linked notebook would be listed

Returns

Either the list of all notebooks within the account or empty list in cases of error or no notebooks presence within the account

4.42.3.36 listAllSavedSearches()

```
QList<SavedSearch> quentier::LocalStorageManager::listAllSavedSearches (
            ErrorString & errorDescription,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListSavedSearchesOrder order = ListSavedSearchesOrder::NoOrder,
            const OrderDirection orderDirection = OrderDirection::Ascending ) const
```

listAllSavedSearches lists all saved searches within the account.

Parameters

errorDescription	Error description if all saved searches could not be listed; otherwise this parameter is untouched
limit	Limit for the max number of saved searches in the result, zero by default which means no limit is set
offset	Number of saved searches to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of saved searches in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder

Returns

Either the list of all saved searches within the account or empty list in case of error or if there are no saved searches within the account

4.42.3.37 listAllSharedNotebooks()

```
QList<SharedNotebook> quentier::LocalStorageManager::listAllSharedNotebooks (
            ErrorString & errorDescription ) const
```

listAllSharedNotebooks attempts to list all shared notebooks within the account.

Parameters

errorDescription	Error description if shared notebooks could not be listed; if no error happens, this parameter is untouched

Returns

Either the list of all shared notebooks within the account or empty list in cases of error or no shared notebooks presence within the account

4.42.3.38 listAllTags()

```
QList<Tag> quentier::LocalStorageManager::listAllTags (
            ErrorString & errorDescription,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListTagsOrder order = ListTagsOrder::NoOrder,
            const OrderDirection orderDirection = OrderDirection::Ascending,
            const QString & linkedNotebookGuid = QString() ) const
```

listAllTags lists all tags within the current user's account.

Parameters

errorDescription	Error description if tags were not listed successfully. In such case the returned list of tags would be empty and error description won't be empty. However, if, for example, the list of tags is empty and error description is empty too, it means the current account does not have any tags created.
limit	Limit for the max number of tags in the result, zero by default which means no limit is set
offset	Number of tags to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of tags in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder
linkedNotebookGuid	If it's null, the method would list tags ignoring their belonging to the current account or to some linked notebook; if it's empty, only the tags from user's own account would be listed; otherwise, only the tags corresponding to the certain linked notebook would be listed

Returns

The list of found tags within the account

4.42.3.39 listAllTagsPerNote()

```
QList<Tag> quentier::LocalStorageManager::listAllTagsPerNote (
            const Note & note,
            ErrorString & errorDescription,
            const ListObjectsOptions & flag = ListObjectsOption::ListAll,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListTagsOrder & order = ListTagsOrder::NoOrder,
            const OrderDirection & orderDirection = OrderDirection::Ascending ) const
```

listAllTagsPerNote lists all tags per given note

Parameters

note	Note for which the list of tags is requested. If it has "remote" Evernote service's guid set, it is used to identify the note in the local storage database. Otherwise its local uid is used for that.
errorDescription	Error description if tags were not listed successfully. In such case the returned list of tags would be empty and error description won't be empty. However, if, for example, the list of tags is empty and error description is empty too, it means the provided note does not have any tags assigned to it.
flag	Input parameter used to set the filter for the desired tags to be listed
limit	Limit for the max number of tags in the result, zero by default which means no limit is set
offset	Number of tags to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of tags in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used;

Returns

The list of found tags per note

4.42.3.40 listLinkedNotebooks()

```
QList<LinkedNotebook> quentier::LocalStorageManager::listLinkedNotebooks (
            const ListObjectsOptions flag,
            ErrorString & errorDescription,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListLinkedNotebooksOrder order = ListLinkedNotebooksOrder::NoOrder,
            const OrderDirection orderDirection = OrderDirection::Ascending ) const
```

listLinkedNotebooks attempts to list linked notebooks within the account according to the specified input flag.

Parameters

flag	Input parameter used to set the filter for the desired linked notebooks to be listed
errorDescription	Error description if linked notebooks within the account could not be listed; if no error happens, this parameter is untouched
limit	Limit for the max number of linked notebooks in the result, zero by default which means no limit is set
offset	Number of linked notebooks to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of linked notebooks in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder

Returns

Either list of linked notebooks within the account conforming to the filter or empty list in cases of error or no linked notebooks conforming to the filter exist within the account

4.42.3.41 listNotebooks()

```
QList<Notebook> quentier::LocalStorageManager::listNotebooks (
            const ListObjectsOptions flag,
            ErrorString & errorDescription,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListNotebooksOrder order = ListNotebooksOrder::NoOrder,
            const OrderDirection orderDirection = OrderDirection::Ascending,
            const QString & linkedNotebookGuid = QString() ) const
```

listNotebooks attempts to list notebooks within the account according to the specified input flag

Parameters

flag	Input parameter used to set the filter for the desired notebooks to be listed
errorDescription	Error description if notebooks within the account could not be listed; if no error happens, this parameter is untouched
limit	The limit for the max number of notebooks in the result, zero by default which means no limit is set
offset	The number of notebooks to skip in the beginning of the result, zero by default
order	Allows to specify a particular ordering of notebooks in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder
linkedNotebookGuid	If it's null, the method would list notebooks ignoring their belonging to the current account or to some linked notebook; if it's empty, only the non-linked notebooks would be listed; otherwise, the only one notebook from the corresponding linked notebook would be listed

Returns

Either the list of notebooks within the account conforming to the filter or empty list in cases of error or no notebooks conforming to the filter exist within the account

4.42.3.42 listNotes()

```
QList<Note> quentier::LocalStorageManager::listNotes (
            const ListObjectsOptions flag,
            const GetNoteOptions options,
            ErrorString & errorDescription,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListNotesOrder order = ListNotesOrder::NoOrder,
            const OrderDirection orderDirection = OrderDirection::Ascending,
            const QString & linkedNotebookGuid = QString() ) const
```

listNotes attempts to list notes within the account according to the specified input flag.

Parameters

flag	Input parameter used to set the filter for the desired notes to be listed
options	Options specifying which optionally includable fields of the note should actually be included
errorDescription	Error description if notes within the account could not be listed; if no error happens, this parameter is untouched
limit	Limit for the max number of notes in the result, zero by default which means no limit is set
offset	Number of notes to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of notes in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder
linkedNotebookGuid	If it's null, notes from both user's own notebooks and linked notebooks would be listed; if it's empty, only the notes from non-linked notebooks would be listed; otherwise, only the notes from the specified linked notebook would be listed

Returns

Either list of notes within the account conforming to the filter or empty list in cases of error or no notes conforming to the filter exist within the account

4.42.3.43 listNotesByLocalUids()

```
QList<Note> quentier::LocalStorageManager::listNotesByLocalUids (
            const QStringList & noteLocalUids,
            const GetNoteOptions options,
            ErrorString & errorDescription,
```

```
            const ListObjectsOptions & flag = ListObjectsOption::ListAll,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListNotesOrder & order = ListNotesOrder::NoOrder,
            const OrderDirection & orderDirection = OrderDirection::Ascending ) const
```

listNotesByLocalUids attempts to list notes given their local uids

The method would only return notes which it managed to find within the local storage i.e. having an invalid local uid in the list won't result in an error, just in the corresponding note not returned within the result

Notes within the result can be additionally filtered with flag parameter

Parameters

noteLocalUids	Local uids of notes to be listed
options	Options specifying which optionally includable fields of the note should actually be included
errorDescription	Error description in case notes could not be listed
flag	Input parameter used to set the filter for the desired notes to be listed
limit	Limit for the max number of notes in the result, zero by default which means no limit is set
offset	Number of notes to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of notes in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used;

Returns

Either list of notes by local uids or empty list in case of error or no notes corresponding to given local uids presence

4.42.3.44 listNotesPerNotebook()

```
QList<Note> quentier::LocalStorageManager::listNotesPerNotebook (
            const Notebook & notebook,
            const GetNoteOptions options,
            ErrorString & errorDescription,
            const ListObjectsOptions & flag = ListObjectsOption::ListAll,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListNotesOrder & order = ListNotesOrder::NoOrder,
            const OrderDirection & orderDirection = OrderDirection::Ascending ) const
```

listNotesPerNotebook attempts to list notes per given notebook

Parameters

notebook	Notebook for which the list of notes is requested. If it has the "remote" Evernote service's guid set, it would be used to identify the notebook in the local storage database, otherwise its local uid would be used
options	Options specifying which optionally includable fields of the note should actually be included
errorDescription	Error description in case notes could not be listed
flag	Input parameter used to set the filter for the desired notes to be listed

Parameters

limit	Limit for the max number of notes in the result, zero by default which means no limit is set
offset	Number of notes to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of notes in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used;

Returns

Either list of notes per notebook or empty list in case of error or no notes presence in the given notebook

4.42.3.45 listNotesPerNotebooksAndTags()

```
QList<Note> quentier::LocalStorageManager::listNotesPerNotebooksAndTags (
            const QStringList & notebookLocalUids,
            const QStringList & tagLocalUids,
            const GetNoteOptions options,
            ErrorString & errorDescription,
            const ListObjectsOptions & flag = ListObjectsOption::ListAll,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListNotesOrder & order = ListNotesOrder::NoOrder,
            const OrderDirection & orderDirection = OrderDirection::Ascending ) const
```

listNotesPerNotebooksAndTags attempts to list notes which are present within one of specified notebooks and are labeled with at least one of specified tags

Parameters

notebookLocalUids	Local uids of notebooks to which the listed notes might belong
tagLocalUids	Local uids of tags with which the listed notes might be labeled
options	Options specifying which optionally includable fields of the note should actually be included
errorDescription	Error description in case notes could not be listed
flag	Input parameter used to set the filter for the desired notes to be listed
limit	Limit for the max number of notes in the result, zero by default which means no limit is set
offset	Number of notes to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of notes in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used;

Returns

Either list of notes per notebooks and tags or empty list in case of error or no notes corresponding to given notebooks and tags presence

4.42.3.46 listNotesPerTag()

```
QList<Note> quentier::LocalStorageManager::listNotesPerTag (
            const Tag & tag,
            const GetNoteOptions options,
            ErrorString & errorDescription,
            const ListObjectsOptions & flag = ListObjectsOption::ListAll,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListNotesOrder & order = ListNotesOrder::NoOrder,
            const OrderDirection & orderDirection = OrderDirection::Ascending ) const
```

listNotesPerTag attempts to list notes labeled with a given tag

Parameters

tag	Tag for which the list of notes labeled with it is requested. If it has the "remote" Evernote service's guid set, it is used to identify the tag in the local storage database, otherwise its local uid is used
options	Options specifying which optionally includable fields of the note should actually be included
errorDescription	Error description in case notes could not be listed
flag	Input parameter used to set the filter for the desired notes to be listed
limit	Limit for the max number of notes in the result, zero by default which means no limit is set
offset	Number of notes to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of notes in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used;

Returns

Either list of notes per tag or empty list in case of error or no notes labeled with the given tag presence

4.42.3.47 listSavedSearches()

```
QList<SavedSearch> quentier::LocalStorageManager::listSavedSearches (
            const ListObjectsOptions flag,
            ErrorString & errorDescription,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListSavedSearchesOrder order = ListSavedSearchesOrder::NoOrder,
            const OrderDirection orderDirection = OrderDirection::Ascending ) const
```

listSavedSearches attempts to list saved searches within the account according to the specified input flag.

Parameters

flag	Input parameter used to set the filter for the desired saved searches to be listed
errorDescription	Error description if saved searches within the account could not be listed; if no error happens, this parameter is untouched
limit	Limit for the max number of saved searches in the result, zero by default which means no limit is set
offset	Number of saved searches to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of saved searches in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder

Returns

Either list of saved searches within the account conforming to the filter or empty list in cases of error or no saved searches conforming to the filter exist within the account

4.42.3.48 listSharedNotebooksPerNotebookGuid()

```
QList<SharedNotebook> quentier::LocalStorageManager::listSharedNotebooksPerNotebookGuid (
            const QString & notebookGuid,
            ErrorString & errorDescription ) const
```

listSharedNotebooksPerNotebookGuid - attempts to list all shared notebooks per given notebook's remote guid (not local uid, it's important).

Parameters

notebookGuid	Remote Evernote service's guid of the notebook for which the shared notebooks are requested
errorDescription	Error description if shared notebooks per notebook guid could not be listed; if no error happens, this parameter is untouched

Returns

Either the list of shared notebooks per notebook guid or empty list in case of error or no shared notebooks presence per given notebook guid

4.42.3.49 listTags()

```
QList<Tag> quentier::LocalStorageManager::listTags (
            const ListObjectsOptions flag,
            ErrorString & errorDescription,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListTagsOrder & order = ListTagsOrder::NoOrder,
            const OrderDirection orderDirection = OrderDirection::Ascending,
            const QString & linkedNotebookGuid = QString() ) const
```

listTags attempts to list tags within the account according to the specified input flag.

Parameters

flag	Input parameter used to set the filter for the desired tags to be listed
errorDescription	Error description if notes within the account could not be listed; if no error happens, this parameter is untouched
limit	Limit for the max number of tags in the result, zero by default which means no limit is set
offset	Number of tags to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of tags in the result, NoOrder by default

Parameters

orderDirection	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder
linkedNotebookGuid	If it's null, the method would list tags ignoring their belonging to the current account or to some linked notebook; if it's empty, only the tags from user's own account would be listed; otherwise, only the tags corresponding to the certain linked notebook would be listed

Returns

Either list of tags within the account conforming to the filter or empty list in cases of error or no tags conforming to the filter exist within the account

4.42.3.50 listTagsWithNoteLocalUids()

```
QList<std::pair<Tag, QStringList> > quentier::LocalStorageManager::listTagsWithNoteLocalUids
(
            const ListObjectsOptions flag,
            ErrorString & errorDescription,
            const size_t limit = 0,
            const size_t offset = 0,
            const ListTagsOrder & order = ListTagsOrder::NoOrder,
            const OrderDirection orderDirection = OrderDirection::Ascending,
            const QString & linkedNotebookGuid = QString() ) const
```

listTagsWithNoteLocalUids attempts to list tags and their corresponding local uids within the account according to the specified input flag

The method is very similar to listTags only for each listed tag it returns the list of note local uids corresponding to notes labeled with the respective tag.

Parameters

flag	Input parameter used to set the filter for the desired tags to be listed
errorDescription	Error description if notes within the account could not be listed; if no error happens, this parameter is untouched
limit	Limit for the max number of tags in the result, zero by default which means no limit is set
offset	Number of tags to skip in the beginning of the result, zero by default
order	Allows to specify particular ordering of tags in the result, NoOrder by default
orderDirection	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder
linkedNotebookGuid	If it's null, the method would list tags ignoring their belonging to the current account or to some linked notebook; if it's empty, only the tags from user's own account would be listed; otherwise, only the tags corresponding to the certain linked notebook would be listed

Returns

Either list of tags and note local uids within the account conforming to the filter or empty list in cases of error or no tags conforming to the filter exist within the account

4.42.3.51 localStorageRequiresUpgrade()

```
bool quentier::LocalStorageManager::localStorageRequiresUpgrade (
            ErrorString & errorDescription )
```

localStorageRequiresUpgrade method checks whether the existing local storage persistence requires to be upgraded. The upgrades may be required sometimes when new version of libquentier is rolled out which changes something in the internals of local storage organization. This method only checks for changes which are backwards incompatible i.e. once the local storage is upgraded, previous version of libquentier won't be able to work with it properly!

NOTE: it is libquentier client code's responsibility to call this method and/or isLocalStorageVersionTooHigh method, libquentier won't call any of these on its own and will just attempt to work with the existing local storage, whatever version it is of. If version is too high, things can fail in most mysterious way, so the client code is obliged to call these methods to ensure the local storage version is checked properly.

Parameters

errorDescription	Textual description of the error if the method was unable to determine whether the local storage requires upgrade, otherwise this parameter is not touched by the method

Returns

True if local storage requires upgrade, false otherwise

4.42.3.52 localStorageVersion()

```
qint32 quentier::LocalStorageManager::localStorageVersion (
            ErrorString & errorDescription )
```

localStorageVersion method fetches the current version of local storage persistence which can be used for informational purposes.

Parameters

errorDescription	Textual description of the error if the method was unable to determine the current version of local storage persistence

Returns

Positive number indication local storage version or negative number in case of error retrieving the local storage version

4.42.3.53 notebookCount()

```
int quentier::LocalStorageManager::notebookCount (
            ErrorString & errorDescription ) const
```

notebookCount returns the number of notebooks currently stored in the local storage database

Parameters

errorDescription	Error description if the number of notebooks could not be returned

Returns

Either non-negative value with the number of notebooks or -1 which means some error has occurred

4.42.3.54 noteCount()

```
int quentier::LocalStorageManager::noteCount (
            ErrorString & errorDescription,
            const NoteCountOptions options = NoteCountOption::IncludeNonDeletedNotes ) const
```

noteCount returns the number of notes currently stored in the local storage database.

Parameters

errorDescription	Error description if the number of notes could not be returned
options	Options clarifying which notes to list; by default only non-deleted notes are listed

Returns

Either non-negative value with the number of notes or -1 which means some error occurred

4.42.3.55 noteCountPerNotebook()

```
int quentier::LocalStorageManager::noteCountPerNotebook (
            const Notebook & notebook,
            ErrorString & errorDescription,
            const NoteCountOptions options = NoteCountOption::IncludeNonDeletedNotes ) const
```

noteCountPerNotebook returns the number of notes currently stored in the local storage database per given notebook.

Parameters

notebook	Notebook for which the number of notes is requested. If its guid is set, it is used to identify the notebook, otherwise its local uid is used
errorDescription	Error description if the number of notes per given notebook could not be returned
options	Options clarifying which notes to list; by default only non-deleted notes are listed

Returns

Either non-negative value with the number of notes per given notebook or -1 which means some error occurred

4.42.3.56 noteCountPerNotebooksAndTags()

```
int quentier::LocalStorageManager::noteCountPerNotebooksAndTags (
            const QStringList & notebookLocalUids,
            const QStringList & tagLocalUids,
            ErrorString & errorDescription,
            const NoteCountOptions options = NoteCountOption::IncludeNonDeletedNotes ) const
```

noteCountPerNotebooksAndTags returns the number of notes currently stored in local storage database belonging to one of notebooks corresponding to given notebook local uids and labeled by at least one of tags corresponding to given tag local uids

Parameters

notebookLocalUids	The list of notebook local uids used for filtering
tagLocalUids	The list of tag local uids used for filtering
errorDescription	Error description if the number of notes per notebooks and tags could not be returned
options	Options clarifying which notes to list; by default only non-deleted notes are listed

Returns

Either non-negative value with the number of notes per given tag or -1 which means some error occurred

4.42.3.57 noteCountPerTag()

```
int quentier::LocalStorageManager::noteCountPerTag (
            const Tag & tag,
            ErrorString & errorDescription,
            const NoteCountOptions options = NoteCountOption::IncludeNonDeletedNotes ) const
```

noteCountPerTag returns the number of notes currently stored in local storage database labeled with given tag.

Parameters

tag	Tag for which the number of notes labeled with it is requested. If its guid is set, it is used to identify the tag, otherwise its local uid is used
errorDescription	Error description if the number of notes per given tag could not be returned
options	Options clarifying which notes to list; by default only non-deleted notes are listed

Returns

Either non-negative value with the number of notes per given tag or -1 which means some error occurred

4.42.3.58 noteCountsPerAllTags()

```
bool quentier::LocalStorageManager::noteCountsPerAllTags (
            QHash< QString, int > & noteCountsPerTagLocalUid,
            ErrorString & errorDescription,
            const NoteCountOptions options = NoteCountOption::IncludeNonDeletedNotes ) const
```

noteCountsPerAllTags returns the number of notes currently stored in local storage database labeled with each tag stored in the local storage database.

Parameters

noteCountsPerTagLocalUid	The result hash: note counts by tag local uids
errorDescription	Error description if the number of notes per all tags could not be returned
options	Options clarifying which notes to list; by default only non-deleted notes are listed

Returns

True if note counts for all tags were computed successfully, false otherwise

4.42.3.59 requiredLocalStoragePatches()

```
QVector<std::shared_ptr<ILocalStoragePatch> > quentier::LocalStorageManager::requiredLocal↩
StoragePatches ( )
```

requiredLocalStoragePatches provides the client code with the list of patches which need to be applied to the current state of local storage in order to bring it to a state compatible with the current version of code. If no patches are required, an empty list of patches is returned.

The client code should apply each patch in the exact order in which they are returned by this method.

Returns

The vector of patches required to be applied to the current local storage version

4.42.3.60 savedSearchCount()

```
int quentier::LocalStorageManager::savedSearchCount (
            ErrorString & errorDescription ) const
```

savedSearchCount returns the number of saved seacrhes currently stored in local storage database.

Parameters

errorDescription	Error description if the number of saved seacrhes could not be returned

Returns

Either non-negative value with the number of saved seacrhes or -1 which means some error occurred

4.42.3.61 switchUser()

```
void quentier::LocalStorageManager::switchUser (
            const Account & account,
            const StartupOptions options = {} )
```

switchUser - switches to another local storage database file associated with the passed in account

If optional "startFromScratch" parameter is set to true (it is false by default), the database file would be erased and only then - opened. If optional "overrideLock" parameter is set to true, the advisory lock set on the database file (if any) would be forcefully removed; otherwise, if this parameter if set to false, the presence of advisory lock on the database file woud cause the method to throw DatabaseLockedException

Parameters

account	The account to which the local storage is to be switched
options	Startup options for the local storage, none enabled by default

4.42.3.62 tagCount()

```
int quentier::LocalStorageManager::tagCount (
            ErrorString & errorDescription ) const
```

tagCount returns the number of non-deleted tags currently stored in the local storage database.

Parameters

errorDescription	Error description if the number of tags could not be returned

Returns

Either non-negative value with the number of tags or -1 which means some error occurred

4.42.3.63 updateEnResource()

```
bool quentier::LocalStorageManager::updateEnResource (
            Resource & resource,
            ErrorString & errorDescription )
```

updateEnResource updates passed in resource in the local storage database.

If the resource has "remote" Evernote service's guid set, it is identified by this guid in the local storage database. If no resource with such guid is found, the local uid is used to identify the resource in the local storage database. If the resource has no guid, the local uid is used to identify it in the local storage database.

Parameters

resource	Resource to be updated; may be changed as a result of the call, automatically filled with local uid and note local uid and/or guid if these were empty before the call
errorDescription	Error description if resource could not be updated

Returns

True if resource was updated successfully, false otherwise

4.42.3.64 updateLinkedNotebook()

```
bool quentier::LocalStorageManager::updateLinkedNotebook (
            const LinkedNotebook & linkedNotebook,
            ErrorString & errorDescription )
```

updateLinkedNotebook updates passd in LinkedNotebook in the local storage database; LinkedNotebook must have "remote" Evernote service's guid set.

Parameters

linkedNotebook	LinkedNotebook to be updated in the local storage database
errorDescription	Error description if linked notebook could not be updated

Returns

True if linked notebook was updated successfully, false otherwise

4.42.3.65 updateNote()

```
bool quentier::LocalStorageManager::updateNote (
            Note & note,
            const UpdateNoteOptions options,
            ErrorString & errorDescription )
```

updateNote updates passed in Note in the local storage database.

If the note has "remote" Evernote service's guid set, it is identified by this guid in the local storage database. If no note with such guid is found, the local uid is used to identify the note in the local storage database. If the note has no guid, the local uid is used to identify it in the local storage database.

A special way in which this method might be used is the update of a note which clears note's guid. This way is special because it imposes certain requirements onto the resources which the note might have. However, it is only relevant if options input parameter has UpdateResourceMetadata flag enabled. The requirements for this special case are as follows:

- each resource should not have noteGuid field set to a non-empty value
- each resource should not have guid field set to a non-empty value as it makes no sense for note without guid i.e. note not synchronized with Evernote to own a resource which has guid i.e. is synchronized with Evernote

Parameters

note	Note to be updated in the local storage database; required to contain either "remote" notebook guid or local notebook uid; may be changed as a result of the call, filled with fields like local uid or notebook guid or local uid if any of these were empty before the call; also tag guids are filled if the note passed in contained only tag local uids and tag local uids are filled if the note passed in contained only tag guids. Bear in mind that after the call the note may not have the representative resources if "updateNoteOptions" input parameter contained no "UpdateResourceMetadata" flag as well as it may not have the representative tags if "UpdateTags" flag was not set
options	Options specifying which optionally updatable fields of the note should actually be updated
errorDescription	Error description if note could not be updated

Returns

True if note was updated successfully, false otherwise

4.42.3.66 updateNotebook()

```
bool quentier::LocalStorageManager::updateNotebook (
            Notebook & notebook,
            ErrorString & errorDescription )
```

updateNotebook updates the passed in Notebook in the local storage database

If the notebook has "remote" Evernote service's guid set, it is identified by this guid in the local storage database. Otherwise it is identified by the local uid.

Parameters

notebook	Notebook to be updated in the local storage database; the object is passed by reference and may be changed as a result of the call (filled with autocompleted fields like local uid if it was empty before the call)
errorDescription	Error description if the notebook could not be updated

Returns

True if the notebook was updated successfully, false otherwise

4.42.3.67 updateSavedSearch()

```
bool quentier::LocalStorageManager::updateSavedSearch (
            SavedSearch & search,
            ErrorString & errorDescription )
```

updateSavedSearch updates passed in SavedSearch in the local storage database.

If search has "remote" Evernote service's guid set, it is identified in the database by this guid. If the saved search has no guid, the local uid is used to identify it in the local storage database.

Parameters

search	SavedSearch filled with values to be updated in the local storage database; may be changed as a result of the call filled local uid if it was empty before the call
errorDescription	Error description if SavedSearch could not be updated

Returns

True if SavedSearch was updated successfully, false otherwise

4.42.3.68 updateTag()

```
bool quentier::LocalStorageManager::updateTag (
            Tag & tag,
            ErrorString & errorDescription )
```

updateTag updates passed in Tag in the local storage database.

If the tag has "remote" Evernote service's guid set, it is identified by this guid in the local storage database. If the tag has no guid, the local uid is used to identify it in the local storage database.

Parameters

tag	Tag filled with values to be updated in the local storage database. Note that it can be changed ∗ as a result of the call: automatically filled with local uid if it was empty before the call
errorDescription	Error description if tag could not be updated

Returns

True if tag was updated successfully, false otherwise

4.42.3.69 updateUser()

```
bool quentier::LocalStorageManager::updateUser (
            const User & user,
            ErrorString & errorDescription )
```

updateUser updates the passed in User object in the local storage database

The table with Users is only involved in operations with notebooks which have "contact" field set which in turn is used with business accounts

Parameters

user	The user to be updated in the local storage database
errorDescription	Error description if the user could not be updated

Returns

True if the user was updated successfully, false otherwise

4.42.3.70 upgradeProgress

```
void quentier::LocalStorageManager::upgradeProgress (
            double progress )  [signal]
```

LocalStorageManager is capable of performing automatic database upgrades if/when it is necessary.

As the database upgrade can be a lengthy operation, this signal is meant to provide some feedback on the progress of the upgrade

Parameters

progress	The value from 0 to 1 denoting the database upgrade progress

4.42.3.71 userCount()

```
int quentier::LocalStorageManager::userCount (
            ErrorString & errorDescription ) const
```

userCount returns the number of non-deleted users currently stored in the local storage database

Parameters

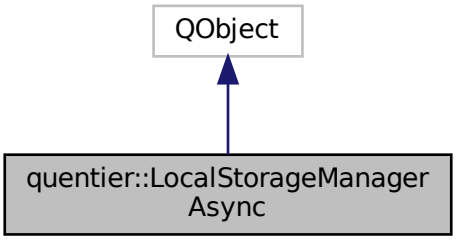
errorDescription	Error description if the number of users could not be returned

Returns

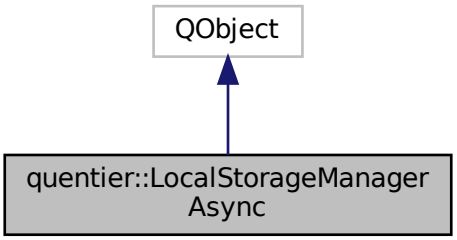
Either non-negative value with the number of users or -1 which means some error has occurred

4.43 quentier::LocalStorageManagerAsync Class Reference

Inheritance diagram for quentier::LocalStorageManagerAsync:



Collaboration diagram for quentier::LocalStorageManagerAsync:



Public Slots

- void **init** ()
- void **onGetUserCountRequest** (QUuid requestId)
- void **onSwitchUserRequest** (Account account, LocalStorageManager::StartupOptions startupOptions, QUuid requestId)
- void **onAddUserRequest** (User user, QUuid requestId)
- void **onUpdateUserRequest** (User user, QUuid requestId)
- void **onFindUserRequest** (User user, QUuid requestId)
- void **onDeleteUserRequest** (User user, QUuid requestId)
- void **onExpungeUserRequest** (User user, QUuid requestId)
- void **onGetNotebookCountRequest** (QUuid requestId)
- void **onAddNotebookRequest** (Notebook notebook, QUuid requestId)
- void **onUpdateNotebookRequest** (Notebook notebook, QUuid requestId)
- void **onFindNotebookRequest** (Notebook notebook, QUuid requestId)
- void **onFindDefaultNotebookRequest** (Notebook notebook, QUuid requestId)
- void **onFindLastUsedNotebookRequest** (Notebook notebook, QUuid requestId)
- void **onFindDefaultOrLastUsedNotebookRequest** (Notebook notebook, QUuid requestId)
- void **onListAllNotebooksRequest** (size_t limit, size_t offset, LocalStorageManager::ListNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QUuid requestId)
- void **onListAllSharedNotebooksRequest** (QUuid requestId)
- void **onListNotebooksRequest** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QUuid requestId)
- void **onListSharedNotebooksPerNotebookGuidRequest** (QString notebookGuid, QUuid requestId)
- void **onExpungeNotebookRequest** (Notebook notebook, QUuid requestId)
- void **onGetLinkedNotebookCountRequest** (QUuid requestId)
- void **onAddLinkedNotebookRequest** (LinkedNotebook linkedNotebook, QUuid requestId)
- void **onUpdateLinkedNotebookRequest** (LinkedNotebook linkedNotebook, QUuid requestId)
- void **onFindLinkedNotebookRequest** (LinkedNotebook linkedNotebook, QUuid requestId)
- void **onListAllLinkedNotebooksRequest** (size_t limit, size_t offset, LocalStorageManager::ListLinkedNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, QUuid requestId)
- void **onListLinkedNotebooksRequest** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListLinkedNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, QUuid requestId)
- void **onExpungeLinkedNotebookRequest** (LinkedNotebook linkedNotebook, QUuid requestId)
- void **onGetNoteCountRequest** (LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **onGetNoteCountPerNotebookRequest** (Notebook notebook, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **onGetNoteCountPerTagRequest** (Tag tag, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **onGetNoteCountsPerAllTagsRequest** (LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **onGetNoteCountPerNotebooksAndTagsRequest** (QStringList notebookLocalUids, QStringList tagLocalUids, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **onAddNoteRequest** (Note note, QUuid requestId)
- void **onUpdateNoteRequest** (Note note, LocalStorageManager::UpdateNoteOptions options, QUuid requestId)
- void **onFindNoteRequest** (Note note, LocalStorageManager::GetNoteOptions options, QUuid requestId)
- void **onListNotesPerNotebookRequest** (Notebook notebook, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QUuid requestId)
- void **onListNotesPerTagRequest** (Tag tag, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QUuid requestId)

- void **onListNotesPerNotebooksAndTagsRequest** (QStringList notebookLocalUids, QStringList tagLocalUids, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QUuid requestId)
- void **onListNotesByLocalUidsRequest** (QStringList noteLocalUids, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QUuid requestId)
- void **onListNotesRequest** (LocalStorageManager::ListObjectsOptions flag, LocalStorageManager::GetNoteOptions options, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QUuid requestId)
- void **onFindNoteLocalUidsWithSearchQuery** (NoteSearchQuery noteSearchQuery, QUuid requestId)
- void **onExpungeNoteRequest** (Note note, QUuid requestId)
- void **onGetTagCountRequest** (QUuid requestId)
- void **onAddTagRequest** (Tag tag, QUuid requestId)
- void **onUpdateTagRequest** (Tag tag, QUuid requestId)
- void **onFindTagRequest** (Tag tag, QUuid requestId)
- void **onListAllTagsPerNoteRequest** (Note note, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QUuid requestId)
- void **onListAllTagsRequest** (size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QUuid requestId)
- void **onListTagsRequest** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QUuid requestId)
- void **onListTagsWithNoteLocalUidsRequest** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QUuid requestId)
- void **onExpungeTagRequest** (Tag tag, QUuid requestId)
- void **onExpungeNotelessTagsFromLinkedNotebooksRequest** (QUuid requestId)
- void **onGetResourceCountRequest** (QUuid requestId)
- void **onAddResourceRequest** (Resource resource, QUuid requestId)
- void **onUpdateResourceRequest** (Resource resource, QUuid requestId)
- void **onFindResourceRequest** (Resource resource, LocalStorageManager::GetResourceOptions options, QUuid requestId)
- void **onExpungeResourceRequest** (Resource resource, QUuid requestId)
- void **onGetSavedSearchCountRequest** (QUuid requestId)
- void **onAddSavedSearchRequest** (SavedSearch search, QUuid requestId)
- void **onUpdateSavedSearchRequest** (SavedSearch search, QUuid requestId)
- void **onFindSavedSearchRequest** (SavedSearch search, QUuid requestId)
- void **onListAllSavedSearchesRequest** (size_t limit, size_t offset, LocalStorageManager::ListSavedSearchesOrder order, LocalStorageManager::OrderDirection orderDirection, QUuid requestId)
- void **onListSavedSearchesRequest** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListSavedSearchesOrder order, LocalStorageManager::OrderDirection orderDirection, QUuid requestId)
- void **onExpungeSavedSearchRequest** (SavedSearch search, QUuid requestId)
- void **onAccountHighUsnRequest** (QString linkedNotebookGuid, QUuid requestId)

Signals

- void **initialized** ()
- void **getUserCountComplete** (int userCount, QUuid requestId)
- void **getUserCountFailed** (ErrorString errorDescription, QUuid requestId)
- void **switchUserComplete** (Account account, QUuid requestId)
- void **switchUserFailed** (Account account, ErrorString errorDescription, QUuid requestId)
- void **addUserComplete** (User user, QUuid requestId)

- void **addUserFailed** (User user, ErrorString errorDescription, QUuid requestId)
- void **updateUserComplete** (User user, QUuid requestId)
- void **updateUserFailed** (User user, ErrorString errorDescription, QUuid requestId)
- void **findUserComplete** (User foundUser, QUuid requestId)
- void **findUserFailed** (User user, ErrorString errorDescription, QUuid requestId)
- void **deleteUserComplete** (User user, QUuid requestId)
- void **deleteUserFailed** (User user, ErrorString errorDescription, QUuid requestId)
- void **expungeUserComplete** (User user, QUuid requestId)
- void **expungeUserFailed** (User user, ErrorString errorDescription, QUuid requestId)
- void **getNotebookCountComplete** (int notebookCount, QUuid requestId)
- void **getNotebookCountFailed** (ErrorString errorDescription, QUuid requestId)
- void **addNotebookComplete** (Notebook notebook, QUuid requestId)
- void **addNotebookFailed** (Notebook notebook, ErrorString errorDescription, QUuid requestId)
- void **updateNotebookComplete** (Notebook notebook, QUuid requestId)
- void **updateNotebookFailed** (Notebook notebook, ErrorString errorDescription, QUuid requestId)
- void **findNotebookComplete** (Notebook foundNotebook, QUuid requestId)
- void **findNotebookFailed** (Notebook notebook, ErrorString errorDescription, QUuid requestId)
- void **findDefaultNotebookComplete** (Notebook foundNotebook, QUuid requestId)
- void **findDefaultNotebookFailed** (Notebook notebook, ErrorString errorDescription, QUuid requestId)
- void **findLastUsedNotebookComplete** (Notebook foundNotebook, QUuid requestId)
- void **findLastUsedNotebookFailed** (Notebook notebook, ErrorString errorDescription, QUuid requestId)
- void **findDefaultOrLastUsedNotebookComplete** (Notebook foundNotebook, QUuid requestId)
- void **findDefaultOrLastUsedNotebookFailed** (Notebook notebook, ErrorString errorDescription, QUuid requestId)
- void **listAllNotebooksComplete** (size_t limit, size_t offset, LocalStorageManager::ListNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QList< Notebook > foundNotebooks, QUuid requestId)
- void **listAllNotebooksFailed** (size_t limit, size_t offset, LocalStorageManager::ListNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, ErrorString errorDescription, QUuid requestId)
- void **listNotebooksComplete** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QList< Notebook > foundNotebooks, QUuid requestId)
- void **listNotebooksFailed** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, ErrorString errorDescription, QUuid requestId)
- void **listAllSharedNotebooksComplete** (QList< SharedNotebook > foundSharedNotebooks, QUuid requestId)
- void **listAllSharedNotebooksFailed** (ErrorString errorDescription, QUuid requestId)
- void **listSharedNotebooksPerNotebookGuidComplete** (QString notebookGuid, QList< SharedNotebook > foundSharedNotebooks, QUuid requestId)
- void **listSharedNotebooksPerNotebookGuidFailed** (QString notebookGuid, ErrorString errorDescription, QUuid requestId)
- void **expungeNotebookComplete** (Notebook notebook, QUuid requestId)
- void **expungeNotebookFailed** (Notebook notebook, ErrorString errorDescription, QUuid requestId)
- void **getLinkedNotebookCountComplete** (int linkedNotebookCount, QUuid requestId)
- void **getLinkedNotebookCountFailed** (ErrorString errorDescription, QUuid requestId)
- void **addLinkedNotebookComplete** (LinkedNotebook linkedNotebook, QUuid requestId)
- void **addLinkedNotebookFailed** (LinkedNotebook linkedNotebook, ErrorString errorDescription, QUuid requestId)
- void **updateLinkedNotebookComplete** (LinkedNotebook linkedNotebook, QUuid requestId)
- void **updateLinkedNotebookFailed** (LinkedNotebook linkedNotebook, ErrorString errorDescription, QUuid requestId)
- void **findLinkedNotebookComplete** (LinkedNotebook foundLinkedNotebook, QUuid requestId)
- void **findLinkedNotebookFailed** (LinkedNotebook linkedNotebook, ErrorString errorDescription, QUuid requestId)

- void **listAllLinkedNotebooksComplete** (size_t limit, size_t offset, LocalStorageManager::ListLinkedNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, QList< LinkedNotebook > foundLinkedNotebooks, QUuid requestId)
- void **listAllLinkedNotebooksFailed** (size_t limit, size_t offset, LocalStorageManager::ListLinkedNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, ErrorString errorDescription, QUuid requestId)
- void **listLinkedNotebooksComplete** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListLinkedNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, QList< LinkedNotebook > foundLinkedNotebooks, QUuid requestId)
- void **listLinkedNotebooksFailed** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListLinkedNotebooksOrder order, LocalStorageManager::OrderDirection orderDirection, ErrorString errorDescription, QUuid requestId)
- void **expungeLinkedNotebookComplete** (LinkedNotebook linkedNotebook, QUuid requestId)
- void **expungeLinkedNotebookFailed** (LinkedNotebook linkedNotebook, ErrorString errorDescription, QUuid requestId)
- void **getNoteCountComplete** (int noteCount, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **getNoteCountFailed** (ErrorString errorDescription, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **getNoteCountPerNotebookComplete** (int noteCount, Notebook notebook, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **getNoteCountPerNotebookFailed** (ErrorString errorDescription, Notebook notebook, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **getNoteCountPerTagComplete** (int noteCount, Tag tag, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **getNoteCountPerTagFailed** (ErrorString errorDescription, Tag tag, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **getNoteCountsPerAllTagsComplete** (QHash< QString, int > noteCountsPerTagLocalUid, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **getNoteCountsPerAllTagsFailed** (ErrorString errorDescription, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **getNoteCountPerNotebooksAndTagsComplete** (int noteCount, QStringList notebookLocalUids, QStringList tagLocalUids, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **getNoteCountPerNotebooksAndTagsFailed** (ErrorString errorDescription, QStringList notebookLocalUids, QStringList tagLocalUids, LocalStorageManager::NoteCountOptions options, QUuid requestId)
- void **addNoteComplete** (Note note, QUuid requestId)
- void **addNoteFailed** (Note note, ErrorString errorDescription, QUuid requestId)
- void **updateNoteComplete** (Note note, LocalStorageManager::UpdateNoteOptions options, QUuid requestId)
- void **updateNoteFailed** (Note note, LocalStorageManager::UpdateNoteOptions options, ErrorString errorDescription, QUuid requestId)
- void **findNoteComplete** (Note foundNote, LocalStorageManager::GetNoteOptions options, QUuid requestId)
- void **findNoteFailed** (Note note, LocalStorageManager::GetNoteOptions options, ErrorString errorDescription, QUuid requestId)
- void **listNotesPerNotebookComplete** (Notebook notebook, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QList< Note > foundNotes, QUuid requestId)
- void **listNotesPerNotebookFailed** (Notebook notebook, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, ErrorString errorDescription, QUuid requestId)
- void **listNotesPerTagComplete** (Tag tag, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QList< Note > foundNotes, QUuid requestId)

- void **listNotesPerTagFailed** (Tag tag, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, ErrorString errorDescription, QUuid requestId)
- void **listNotesPerNotebooksAndTagsComplete** (QStringList notebookLocalUids, QStringList tagLocalUids, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QList< Note > foundNotes, QUuid requestId)
- void **listNotesPerNotebooksAndTagsFailed** (QStringList notebookLocalUids, QStringList tagLocalUids, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, ErrorString errorDescription, QUuid requestId)
- void **listNotesByLocalUidsComplete** (QStringList noteLocalUids, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QList< Note > foundNotes, QUuid requestId)
- void **listNotesByLocalUidsFailed** (QStringList noteLocalUids, LocalStorageManager::GetNoteOptions options, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, ErrorString errorDescription, QUuid requestId)
- void **listNotesComplete** (LocalStorageManager::ListObjectsOptions flag, LocalStorageManager::GetNoteOptions options, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QList< Note > foundNotes, QUuid requestId)
- void **listNotesFailed** (LocalStorageManager::ListObjectsOptions flag, LocalStorageManager::GetNoteOptions options, size_t limit, size_t offset, LocalStorageManager::ListNotesOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, ErrorString errorDescription, QUuid requestId)
- void **findNoteLocalUidsWithSearchQueryComplete** (QStringList noteLocalUids, NoteSearchQuery noteSearchQuery, QUuid requestId)
- void **findNoteLocalUidsWithSearchQueryFailed** (NoteSearchQuery noteSearchQuery, ErrorString errorDescription, QUuid requestId)
- void **expungeNoteComplete** (Note note, QUuid requestId)
- void **expungeNoteFailed** (Note note, ErrorString errorDescription, QUuid requestId)
- void **noteMovedToAnotherNotebook** (QString noteLocalUid, QString previousNotebookLocalUid, QString newNotebookLocalUid)
- void **noteTagListChanged** (QString noteLocalUid, QStringList previousNoteTagLocalUids, QStringList newNoteTagLocalUids)
- void **getTagCountComplete** (int tagCount, QUuid requestId)
- void **getTagCountFailed** (ErrorString errorDescription, QUuid requestId)
- void **addTagComplete** (Tag tag, QUuid requestId)
- void **addTagFailed** (Tag tag, ErrorString errorDescription, QUuid requestId)
- void **updateTagComplete** (Tag tag, QUuid requestId)
- void **updateTagFailed** (Tag tag, ErrorString errorDescription, QUuid requestId)
- void **linkTagWithNoteComplete** (Tag tag, Note note, QUuid requestId)
- void **linkTagWithNoteFailed** (Tag tag, Note note, ErrorString errorDescription, QUuid requestId)
- void **findTagComplete** (Tag tag, QUuid requestId)
- void **findTagFailed** (Tag tag, ErrorString errorDescription, QUuid requestId)
- void **listAllTagsPerNoteComplete** (QList< Tag > foundTags, Note note, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QUuid requestId)
- void **listAllTagsPerNoteFailed** (Note note, LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, ErrorString errorDescription, QUuid requestId)
- void **listAllTagsComplete** (size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QList< Tag > foundTags, QUuid requestId)

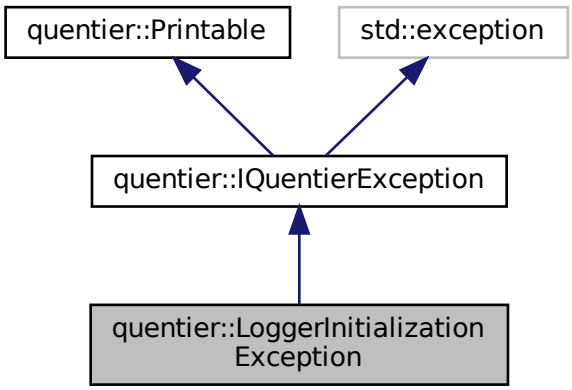
- void **listAllTagsFailed** (size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, ErrorString errorDescription, QUuid requestId)
- void **listTagsComplete** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QList< Tag > foundTags, QUuid requestId=QUuid())
- void **listTagsFailed** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, ErrorString errorDescription, QUuid requestId)
- void **listTagsWithNoteLocalUidsComplete** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, QList< std::pair< Tag, QStringList >> foundTags, QUuid requestId)
- void **listTagsWithNoteLocalUidsFailed** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order, LocalStorageManager::OrderDirection orderDirection, QString linkedNotebookGuid, ErrorString errorDescription, QUuid requestId)
- void **expungeTagComplete** (Tag tag, QStringList expungedChildTagLocalUids, QUuid requestId)
- void **expungeTagFailed** (Tag tag, ErrorString errorDescription, QUuid requestId)
- void **expungeNotelessTagsFromLinkedNotebooksComplete** (QUuid requestId)
- void **expungeNotelessTagsFromLinkedNotebooksFailed** (ErrorString errorDescription, QUuid requestId)
- void **getResourceCountComplete** (int resourceCount, QUuid requestId)
- void **getResourceCountFailed** (ErrorString errorDescription, QUuid requestId)
- void **addResourceComplete** (Resource resource, QUuid requestId)
- void **addResourceFailed** (Resource resource, ErrorString errorDescription, QUuid requestId)
- void **updateResourceComplete** (Resource resource, QUuid requestId)
- void **updateResourceFailed** (Resource resource, ErrorString errorDescription, QUuid requestId)
- void **findResourceComplete** (Resource resource, LocalStorageManager::GetResourceOptions options, QUuid requestId)
- void **findResourceFailed** (Resource resource, LocalStorageManager::GetResourceOptions options, ErrorString errorDescription, QUuid requestId)
- void **expungeResourceComplete** (Resource resource, QUuid requestId)
- void **expungeResourceFailed** (Resource resource, ErrorString errorDescription, QUuid requestId)
- void **getSavedSearchCountComplete** (int savedSearchCount, QUuid requestId)
- void **getSavedSearchCountFailed** (ErrorString errorDescription, QUuid requestId)
- void **addSavedSearchComplete** (SavedSearch search, QUuid requestId)
- void **addSavedSearchFailed** (SavedSearch search, ErrorString errorDescription, QUuid requestId)
- void **updateSavedSearchComplete** (SavedSearch search, QUuid requestId)
- void **updateSavedSearchFailed** (SavedSearch search, ErrorString errorDescription, QUuid requestId)
- void **findSavedSearchComplete** (SavedSearch search, QUuid requestId)
- void **findSavedSearchFailed** (SavedSearch search, ErrorString errorDescription, QUuid requestId)
- void **listAllSavedSearchesComplete** (size_t limit, size_t offset, LocalStorageManager::ListSavedSearchesOrder order, LocalStorageManager::OrderDirection orderDirection, QList< SavedSearch > foundSearches, QUuid requestId)
- void **listAllSavedSearchesFailed** (size_t limit, size_t offset, LocalStorageManager::ListSavedSearchesOrder order, LocalStorageManager::OrderDirection orderDirection, ErrorString errorDescription, QUuid requestId)
- void **listSavedSearchesComplete** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListSavedSearchesOrder order, LocalStorageManager::OrderDirection orderDirection, QList< SavedSearch > foundSearches, QUuid requestId)
- void **listSavedSearchesFailed** (LocalStorageManager::ListObjectsOptions flag, size_t limit, size_t offset, LocalStorageManager::ListSavedSearchesOrder order, LocalStorageManager::OrderDirection orderDirection, ErrorString errorDescription, QUuid requestId)
- void **expungeSavedSearchComplete** (SavedSearch search, QUuid requestId)
- void **expungeSavedSearchFailed** (SavedSearch search, ErrorString errorDescription, QUuid requestId)
- void **accountHighUsnComplete** (qint32 usn, QString linkedNotebookGuid, QUuid requestId)
- void **accountHighUsnFailed** (QString linkedNotebookGuid, ErrorString errorDescription, QUuid requestId)

Public Member Functions

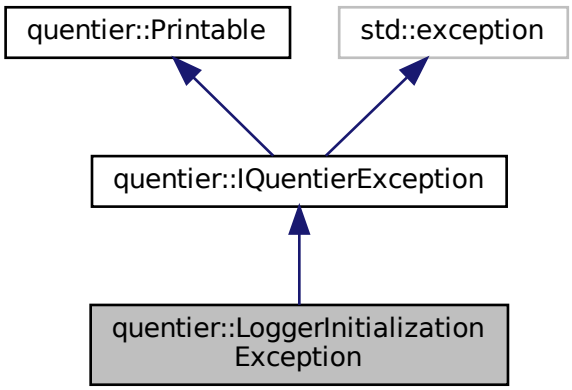
- **LocalStorageManagerAsync** (const Account &account, LocalStorageManager::StartupOptions options={}, QObject *parent=nullptr)
- void **setUseCache** (const bool useCache)
- const LocalStorageCacheManager * **localStorageCacheManager** () const
- bool **installCacheExpiryFunction** (const ILocalStorageCacheExpiryChecker &checker)
- const LocalStorageManager * **localStorageManager** () const
- LocalStorageManager * **localStorageManager** ()

4.44 quentier::LoggerInitializationException Class Reference

Inheritance diagram for quentier::LoggerInitializationException:



Collaboration diagram for quentier::LoggerInitializationException:



Public Member Functions

- **LoggerInitializationException** (const ErrorString &message)

Protected Member Functions

- const QString **exceptionDisplayName** () const override

4.45 quentier::LRUCache< Key, Value, Allocator > Class Template Reference

Public Types

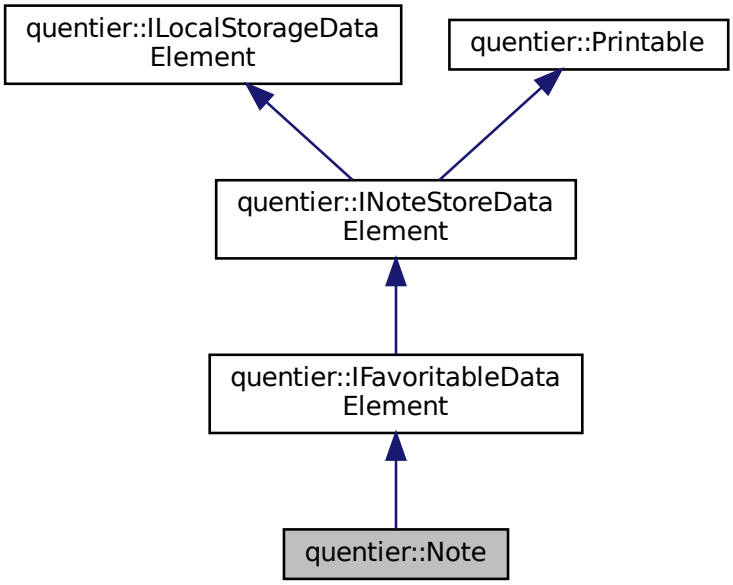
- using **key_type** = Key
- using **mapped_type** = Value
- using **allocator_type** = Allocator
- using **value_type** = std::pair< key_type, mapped_type >
- using **container_type** = std::list< value_type, allocator_type >
- using **size_type** = typename container_type::size_type
- using **difference_type** = typename container_type::difference_type
- using **iterator** = typename container_type::iterator
- using **const_iterator** = typename container_type::const_iterator
- using **reverse_iterator** = std::reverse_iterator< iterator >
- using **const_reverse_iterator** = std::reverse_iterator< const_iterator >
- using **reference** = value_type &
- using **const_reference** = const value_type &
- using **pointer** = typename std::allocator_traits< allocator_type >::pointer
- using **const_pointer** = typename std::allocator_traits< allocator_type >::const_pointer

Public Member Functions

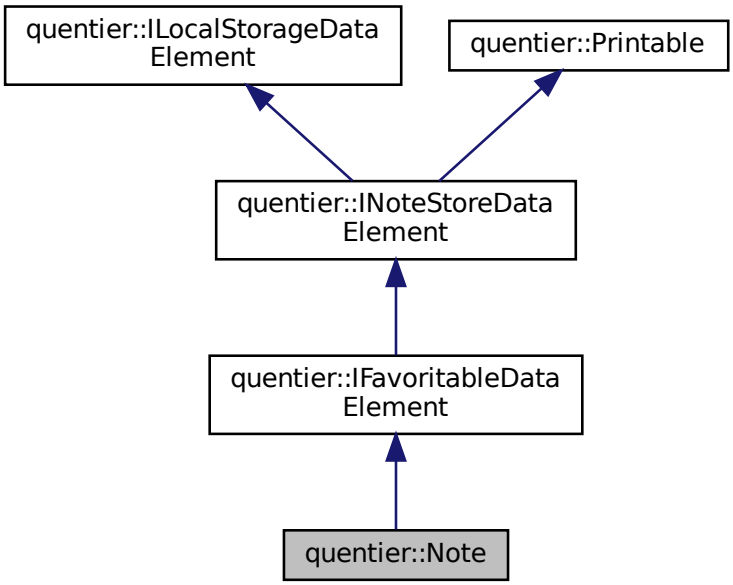
- **LRUCache** (const size_t maxSize=100)
- iterator **begin** ()
- const_iterator **begin** () const
- reverse_iterator **rbegin** ()
- const_reverse_iterator **rbegin** () const
- iterator **end** ()
- const_iterator **end** () const
- reverse_iterator **rend** ()
- const_reverse_iterator **rend** () const
- bool **empty** () const
- size_t **size** () const
- size_t **max_size** () const
- void **clear** ()
- void **put** (const key_type &key, const mapped_type &value)
- const mapped_type * **get** (const key_type &key) const
- bool **exists** (const key_type &key)
- bool **remove** (const key_type &key)
- void **setMaxSize** (const size_t maxSize)

4.46 quentier::Note Class Reference

Inheritance diagram for quentier::Note:



Collaboration diagram for quentier::Note:



Public Member Functions

- **Note** (const Note &other)
- **Note** (Note &&other)
- Note & **operator=** (const Note &other)
- Note & **operator=** (Note &&other)
- **Note** (const qevercloud::Note &other)
- Note & **operator=** (const qevercloud::Note &other)
- bool **operator==** (const Note &other) const
- bool **operator!=** (const Note &other) const
- const qevercloud::Note & **qevercloudNote** () const
- qevercloud::Note & **qevercloudNote** ()
- virtual bool **hasGuid** () const override
- virtual const QString & **guid** () const override
- virtual void **setGuid** (const QString &guid) override
- virtual bool **hasUpdateSequenceNumber** () const override
- virtual qint32 **updateSequenceNumber** () const override
- virtual void **setUpdateSequenceNumber** (const qint32 usn) override
- virtual void **clear** () override
- virtual bool **checkParameters** (ErrorString &errorDescription) const override
- bool **hasTitle** () const
- const QString & **title** () const
- void **setTitle** (const QString &title)
- bool **hasContent** () const
- const QString & **content** () const
- void **setContent** (const QString &content)
- bool **hasContentHash** () const
- const QByteArray & **contentHash** () const
- void **setContentHash** (const QByteArray &contentHash)
- bool **hasContentLength** () const
- qint32 **contentLength** () const
- void **setContentLength** (const qint32 length)
- bool **hasCreationTimestamp** () const
- qint64 **creationTimestamp** () const
- void **setCreationTimestamp** (const qint64 timestamp)
- bool **hasModificationTimestamp** () const
- qint64 **modificationTimestamp** () const
- void **setModificationTimestamp** (const qint64 timestamp)
- bool **hasDeletionTimestamp** () const
- qint64 **deletionTimestamp** () const
- void **setDeletionTimestamp** (const qint64 timestamp)
- bool **hasActive** () const
- bool **active** () const
- void **setActive** (const bool active)
- bool **hasNotebookGuid** () const
- const QString & **notebookGuid** () const
- void **setNotebookGuid** (const QString &guid)
- bool **hasNotebookLocalUid** () const
- const QString & **notebookLocalUid** () const
- void **setNotebookLocalUid** (const QString &notebookLocalUid)
- bool **hasTagGuids** () const
- const QStringList **tagGuids** () const
- void **setTagGuids** (const QStringList &guids)
- void **addTagGuid** (const QString &guid)
- void **removeTagGuid** (const QString &guid)

- bool **hasTagLocalUids** () const
- const QStringList & **tagLocalUids** () const
- void **setTagLocalUids** (const QStringList &localUids)
- void **addTagLocalUid** (const QString &localUid)
- void **removeTagLocalUid** (const QString &localUid)
- bool **hasResources** () const
- int **numResources** () const
- QList< Resource > **resources** () const
- void **setResources** (const QList< Resource > &resources)
- void **addResource** (const Resource &resource)
- bool **updateResource** (const Resource &resource)
- bool **removeResource** (const Resource &resource)
- bool **hasNoteAttributes** () const
- const qevercloud::NoteAttributes & **noteAttributes** () const
- qevercloud::NoteAttributes & **noteAttributes** ()
- void **clearNoteAttributes** ()
- bool **hasSharedNotes** () const
- QList< SharedNote > **sharedNotes** () const
- void **setSharedNotes** (const QList< SharedNote > &sharedNotes)
- void **addSharedNote** (const SharedNote &sharedNote)
- bool **updateSharedNote** (const SharedNote &sharedNote)
- bool **removeSharedNote** (const SharedNote &sharedNote)
- bool **hasNoteRestrictions** () const
- const qevercloud::NoteRestrictions & **noteRestrictions** () const
- qevercloud::NoteRestrictions & **noteRestrictions** ()
- void **setNoteRestrictions** (qevercloud::NoteRestrictions &&restrictions)
- bool **hasNoteLimits** () const
- const qevercloud::NoteLimits & **noteLimits** () const
- qevercloud::NoteLimits & **noteLimits** ()
- void **setNoteLimits** (qevercloud::NoteLimits &&limits)
- QByteArray **thumbnailData** () const
- void **setThumbnailData** (const QByteArray &thumbnailData)
- bool **isInkNote** () const
- QString **plainText** (ErrorString *pErrorMessage=nullptr) const
- QStringList **listOfWords** (ErrorString *pErrorMessage=nullptr) const
- std::pair< QString, QStringList > **plainTextAndListOfWords** (ErrorString *pErrorMessage=nullptr) const
- bool **containsCheckedTodo** () const
- bool **containsUncheckedTodo** () const
- bool **containsTodo** () const
- bool **containsEncryption** () const
- virtual QTextStream & **print** (QTextStream &strm) const override

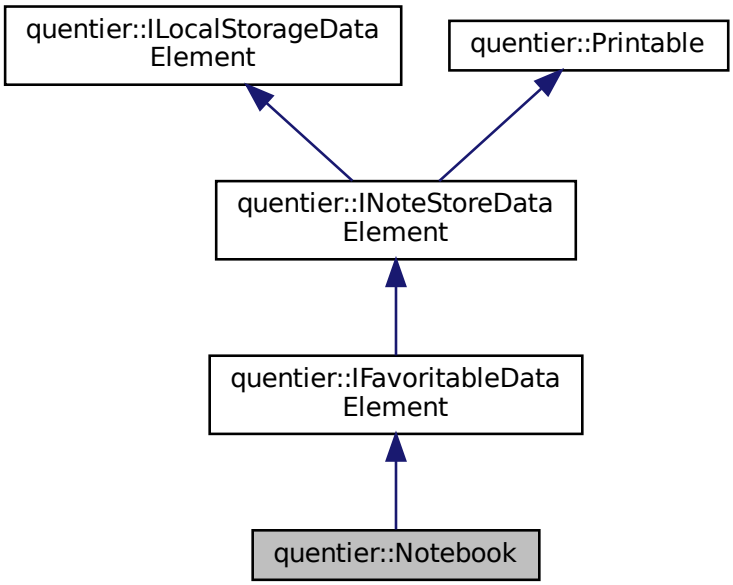
Static Public Member Functions

- static bool **validateTitle** (const QString &title, ErrorString *pErrorDescription=nullptr)

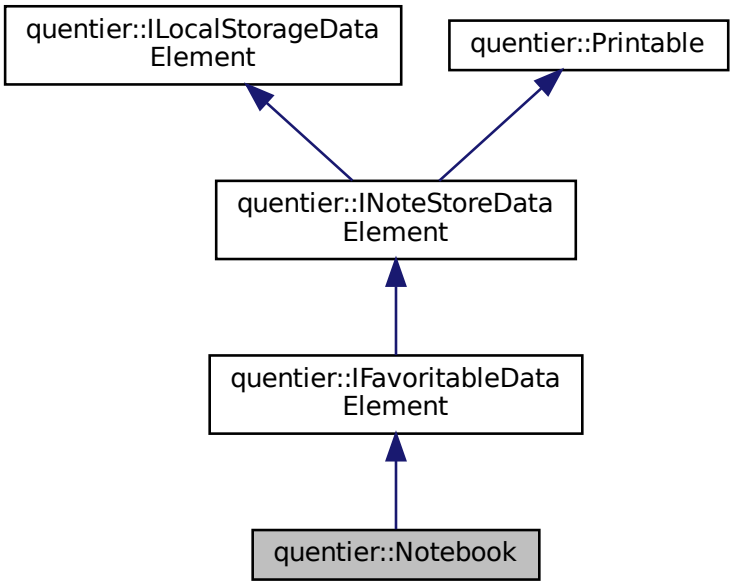
Additional Inherited Members

4.47 quentier::Notebook Class Reference

Inheritance diagram for quentier::Notebook:



Collaboration diagram for quentier::Notebook:



Public Member Functions

- **Notebook** (const Notebook &other)
- **Notebook** (Notebook &&other)
- Notebook & **operator=** (const Notebook &other)
- Notebook & **operator=** (Notebook &&other)
- **Notebook** (const qevercloud::Notebook &other)
- **Notebook** (qevercloud::Notebook &&other)
- Notebook & **operator=** (const qevercloud::Notebook &other)
- Notebook & **operator=** (qevercloud::Notebook &&other)
- bool **operator==** (const Notebook &other) const
- bool **operator!=** (const Notebook &other) const
- const qevercloud::Notebook & **qevercloudNotebook** () const
- qevercloud::Notebook & **qevercloudNotebook** ()
- virtual void **clear** () override
- virtual bool **hasGuid** () const override
- virtual const QString & **guid** () const override
- virtual void **setGuid** (const QString &guid) override
- virtual bool **hasUpdateSequenceNumber** () const override
- virtual qint32 **updateSequenceNumber** () const override
- virtual void **setUpdateSequenceNumber** (const qint32 usn) override
- virtual bool **checkParameters** (ErrorString &errorDescription) const override
- bool **hasName** () const
- const QString & **name** () const
- void **setName** (const QString &name)
- bool **isDefaultNotebook** () const
- void **setDefaultNotebook** (const bool defaultNotebook)

- bool **hasLinkedNotebookGuid** () const
- const QString & **linkedNotebookGuid** () const
- void **setLinkedNotebookGuid** (const QString &linkedNotebookGuid)
- bool **hasCreationTimestamp** () const
- qint64 **creationTimestamp** () const
- void **setCreationTimestamp** (const qint64 timestamp)
- bool **hasModificationTimestamp** () const
- qint64 **modificationTimestamp** () const
- void **setModificationTimestamp** (const qint64 timestamp)
- bool **hasPublishingUri** () const
- const QString & **publishingUri** () const
- void **setPublishingUri** (const QString &uri)
- bool **hasPublishingOrder** () const
- qint8 **publishingOrder** () const
- void **setPublishingOrder** (const qint8 order)
- bool **hasPublishingAscending** () const
- bool **isPublishingAscending** () const
- void **setPublishingAscending** (const bool ascending)
- bool **hasPublishingPublicDescription** () const
- const QString & **publishingPublicDescription** () const
- void **setPublishingPublicDescription** (const QString &publishingPublicDescription)
- bool **hasPublished** () const
- bool **isPublished** () const
- void **setPublished** (const bool published)
- bool **hasStack** () const
- const QString & **stack** () const
- void **setStack** (const QString &stack)
- bool **hasSharedNotebooks** ()
- QList< SharedNotebook > **sharedNotebooks** () const
- void **setSharedNotebooks** (QList< qevercloud::SharedNotebook > sharedNotebooks)
- void **setSharedNotebooks** (QList< SharedNotebook > &&notebooks)
- void **addSharedNotebook** (const SharedNotebook &sharedNotebook)
- void **removeSharedNotebook** (const SharedNotebook &sharedNotebook)
- bool **hasBusinessNotebookDescription** () const
- const QString & **businessNotebookDescription** () const
- void **setBusinessNotebookDescription** (const QString &businessNotebookDescription)
- bool **hasBusinessNotebookPrivilegeLevel** () const
- qint8 **businessNotebookPrivilegeLevel** () const
- void **setBusinessNotebookPrivilegeLevel** (const qint8 privilegeLevel)
- bool **hasBusinessNotebookRecommended** () const
- bool **isBusinessNotebookRecommended** () const
- void **setBusinessNotebookRecommended** (const bool recommended)
- bool **hasContact** () const
- const User **contact** () const
- void **setContact** (const User &contact)
- bool **isLastUsed** () const
- void **setLastUsed** (const bool lastUsed)
- bool **canReadNotes** () const
- void **setCanReadNotes** (const bool canReadNotes)
- bool **canCreateNotes** () const
- void **setCanCreateNotes** (const bool canCreateNotes)
- bool **canUpdateNotes** () const
- void **setCanUpdateNotes** (const bool canUpdateNotes)
- bool **canExpungeNotes** () const
- void **setCanExpungeNotes** (const bool canExpungeNotes)

- bool **canShareNotes** () const
- void **setCanShareNotes** (const bool canShareNotes)
- bool **canEmailNotes** () const
- void **setCanEmailNotes** (const bool canEmailNotes)
- bool **canSendMessageToRecipients** () const
- void **setCanSendMessageToRecipients** (const bool canSendMessageToRecipients)
- bool **canUpdateNotebook** () const
- void **setCanUpdateNotebook** (const bool canUpdateNotebook)
- bool **canExpungeNotebook** () const
- void **setCanExpungeNotebook** (const bool canExpungeNotebook)
- bool **canSetDefaultNotebook** () const
- void **setCanSetDefaultNotebook** (const bool canSetDefaultNotebook)
- bool **canSetNotebookStack** () const
- void **setCanSetNotebookStack** (const bool canSetNotebookStack)
- bool **canPublishToPublic** () const
- void **setCanPublishToPublic** (const bool canPublishToPublic)
- bool **canPublishToBusinessLibrary** () const
- void **setCanPublishToBusinessLibrary** (const bool canPublishToBusinessLibrary)
- bool **canCreateTags** () const
- void **setCanCreateTags** (const bool canCreateTags)
- bool **canUpdateTags** () const
- void **setCanUpdateTags** (const bool canUpdateTags)
- bool **canExpungeTags** () const
- void **setCanExpungeTags** (const bool canExpungeTags)
- bool **canSetParentTag** () const
- void **setCanSetParentTag** (const bool canSetParentTag)
- bool **canCreateSharedNotebooks** () const
- void **setCanCreateSharedNotebooks** (const bool canCreateSharedNotebooks)
- bool **canShareNotesWithBusiness** () const
- void **setCanShareNotesWithBusiness** (const bool canShareNotesWithBusiness)
- bool **canRenameNotebook** () const
- void **setCanRenameNotebook** (const bool canRenameNotebook)
- bool **hasUpdateWhichSharedNotebookRestrictions** () const
- qint8 **updateWhichSharedNotebookRestrictions** () const
- void **setUpdateWhichSharedNotebookRestrictions** (const qint8 which)
- bool **hasExpungeWhichSharedNotebookRestrictions** () const
- qint8 **expungeWhichSharedNotebookRestrictions** () const
- void **setExpungeWhichSharedNotebookRestrictions** (const qint8 which)
- bool **hasRestrictions** () const
- const qevercloud::NotebookRestrictions & **restrictions** () const
- void **setNotebookRestrictions** (qevercloud::NotebookRestrictions &&restrictions)
- bool **hasRecipientReminderNotifyEmail** () const
- bool **recipientReminderNotifyEmail** () const
- void **setRecipientReminderNotifyEmail** (const bool notifyEmail)
- bool **hasRecipientReminderNotifyInApp** () const
- bool **recipientReminderNotifyInApp** () const
- void **setRecipientReminderNotifyInApp** (const bool notifyInApp)
- bool **hasRecipientInMyList** () const
- bool **recipientInMyList** () const
- void **setRecipientInMyList** (const bool inMyList)
- bool **hasRecipientStack** () const
- const QString & **recipientStack** () const
- void **setRecipientStack** (const QString &recipientString)
- bool **hasRecipientSettings** () const
- const qevercloud::NotebookRecipientSettings & **recipientSettings** () const
- void **setNotebookRecipientSettings** (qevercloud::NotebookRecipientSettings &&settings)
- virtual QTextStream & **print** (QTextStream &strm) const override

Static Public Member Functions

- static bool **validateName** (const QString &name, ErrorString *pErrorDescription=nullptr)

Additional Inherited Members

4.48 quentier::ENMLConverter::NoteContentToHtmlExtraData Struct Reference

Public Attributes

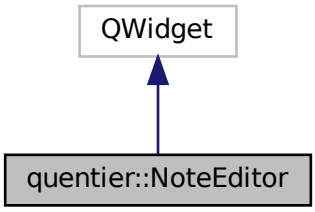
- quint64 **m_numEnToDoNodes** = 0
- quint64 **m_numHyperlinkNodes** = 0
- quint64 **m_numEnCryptNodes** = 0
- quint64 **m_numEnDecryptedNodes** = 0

4.49 quentier::NoteEditor Class Reference

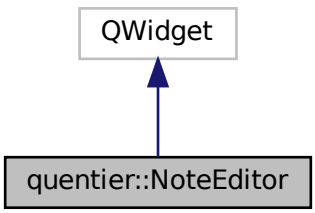
The NoteEditor class is a widget encapsulating all the functionality necessary for showing and editing notes.

`#include <NoteEditor.h>`

Inheritance diagram for quentier::NoteEditor:



Collaboration diagram for quentier::NoteEditor:



Public Slots

- void convertToNote ()
- void saveNoteToLocalStorage ()
- void setNoteTitle (const QString &noteTitle)
- void setTagIds (const QStringList &tagLocalUids, const QStringList &tagGuids)
- void **undo** ()
- void **redo** ()
- void **cut** ()
- void **copy** ()
- void **paste** ()
- void **pasteUnformatted** ()
- void **selectAll** ()
- void **formatSelectionAsSourceCode** ()
- void **fontMenu** ()
- void **textBold** ()
- void **textItalic** ()
- void **textUnderline** ()
- void **textStrikethrough** ()
- void **textHighlight** ()
- void **alignLeft** ()
- void **alignCenter** ()
- void **alignRight** ()
- void **alignFull** ()
- void **findNext** (const QString &text, const bool matchCase) const
- void **findPrevious** (const QString &text, const bool matchCase) const
- void **replace** (const QString &textToReplace, const QString &replacementText, const bool matchCase)
- void **replaceAll** (const QString &textToReplace, const QString &replacementText, const bool matchCase)
- void **insertToDoCheckbox** ()
- void **insertInAppNoteLink** (const QString &userId, const QString &shardId, const QString &noteGuid, const QString &linkText)
- void **setSpellcheck** (const bool enabled)
- void **setFont** (const QFont &font)
- void **setFontHeight** (const int height)
- void **setFontColor** (const QColor &color)
- void **setBackgroundColor** (const QColor &color)
- void setDefaultPalette (const QPalette &pal)
- void setDefaultFont (const QFont &font)
- void **insertHorizontalLine** ()
- void **increaseFontSize** ()
- void **decreaseFontSize** ()
- void **increaseIndentation** ()
- void **decreaseIndentation** ()
- void **insertBulletedList** ()
- void **insertNumberedList** ()
- void **insertTableDialog** ()
- void **insertFixedWidthTable** (const int rows, const int columns, const int widthInPixels)
- void **insertRelativeWidthTable** (const int rows, const int columns, const double relativeWidth)
- void **insertTableRow** ()
- void **insertTableColumn** ()
- void **removeTableRow** ()
- void **removeTableColumn** ()
- void **addAttachmentDialog** ()
- void **saveAttachmentDialog** (const QByteArray &resourceHash)
- void **saveAttachmentUnderCursor** ()

- void **openAttachment** (const QByteArray &resourceHash)
- void **openAttachmentUnderCursor** ()
- void **copyAttachment** (const QByteArray &resourceHash)
- void **copyAttachmentUnderCursor** ()
- void **encryptSelectedText** ()
- void **decryptEncryptedTextUnderCursor** ()
- void **editHyperlinkDialog** ()
- void **copyHyperlink** ()
- void **removeHyperlink** ()
- void **onNoteLoadCancelled** ()

Signals

- void contentChanged ()

  contentChanged signal is emitted when the note's content (text) gets modified via manual editing (i.e. not any action like paste or cut)
- void noteAndNotebookFoundInLocalStorage (Note note, Notebook notebook)

  noteAndNotebookFoundInLocalStorage signal is emitted when note and its corresponding notebook were found within the local storage right before the note editor starts to load the note into the editor
- void noteNotFound (QString noteLocalUid)

  noteNotFound signal is emitted when the note could not be found within the local storage by the provided local uid
- void noteDeleted (QString noteLocalUid)

  noteDeleted signal is emitted when the note displayed within the note editor is deleted. The note editor stops displaying the note in this case shortly after emitting this signal
- void noteModified ()

  noteModified signal is emitted when the note's content within the editor gets modified via some way - either via manual editing or via some action (like paste or cut)
- void notifyError (ErrorString error)

  notifyError signal is emitted when NoteEditor encounters some problem worth letting the user to know about
- void inAppNoteLinkClicked (QString userId, QString shardId, QString noteGuid)

  inAppNoteLinkClicked signal is emitted when the in-app note link is clicked within the note editor
- void inAppNoteLinkPasteRequested (QString url, QString userId, QString shardId, QString noteGuid)
- void **convertedToNote** (Note note)
- void **cantConvertToNote** (ErrorString error)
- void **noteEditorHtmlUpdated** (QString html)
- void **currentNoteChanged** (Note note)
- void **spellCheckerNotReady** ()
- void **spellCheckerReady** ()
- void **noteLoaded** ()
- void noteSavedToLocalStorage (QString noteLocalUid)

  noteSavedToLocalStorage signal is emitted when the note has been saved within the local storage. NoteEditor doesn't do this on its own unless it's explicitly asked to do this via invoking its saveNoteToLocalStorage slot
- void failedToSaveNoteToLocalStorage (ErrorString errorDescription, QString noteLocalUid)

  failedToSaveNoteToLocalStorage signal is emitted in case of failure to save the note to local storage
- void **textBoldState** (bool state)
- void **textItalicState** (bool state)
- void **textUnderlineState** (bool state)
- void **textStrikethroughState** (bool state)
- void **textAlignLeftState** (bool state)
- void **textAlignCenterState** (bool state)
- void **textAlignRightState** (bool state)
- void **textAlignFullState** (bool state)
- void **textInsideOrderedListState** (bool state)

- void **textInsideUnorderedListState** (bool state)
- void **textInsideTableState** (bool state)
- void **textFontFamilyChanged** (QString fontFamily)
- void **textFontSizeChanged** (int fontSize)
- void **insertTableDialogRequested** ()

Public Member Functions

- **NoteEditor** (QWidget *parent=nullptr, Qt::WindowFlags flags={})
- void initialize (LocalStorageManagerAsync &localStorageManager, SpellChecker &spellChecker, const Account &account, QThread *pBackgroundJobsThread=nullptr)
- INoteEditorBackend * backend ()
- void setBackend (INoteEditorBackend *backend)
- void setAccount (const Account &account)
- const QUndoStack * undoStack () const
- void setUndoStack (QUndoStack *pUndoStack)
- void setInitialPageHtml (const QString &html)
- void setNoteNotFoundPageHtml (const QString &html)
- void setNoteDeletedPageHtml (const QString &html)
- void setNoteLoadingPageHtml (const QString &html)
- QString currentNoteLocalUid () const
- void setCurrentNoteLocalUid (const QString &noteLocalUid)
- void clear ()
- bool isModified () const
- bool isEditorPageModified () const
- bool isNoteLoaded () const
- qint64 idleTime () const
- void setFocus ()
- QString **selectedText** () const
- bool **hasSelection** () const
- bool **spellCheckEnabled** () const
- bool **print** (QPrinter &printer, ErrorString &errorDescription)
- bool **exportToPdf** (const QString &absoluteFilePath, ErrorString &errorDescription)
- bool **exportToEnex** (const QStringList &tagNames, QString &enex, ErrorString &errorDescription)
- QPalette defaultPalette () const
- const QFont * defaultFont () const

Protected Member Functions

- virtual void **dragMoveEvent** (QDragMoveEvent *pEvent) override
- virtual void **dropEvent** (QDropEvent *pEvent) override

4.49.1 Detailed Description

The NoteEditor class is a widget encapsulating all the functionality necessary for showing and editing notes.

4.49.2 Member Function Documentation

4.49.2.1 backend()

```
INoteEditorBackend* quentier::NoteEditor::backend ( )
```

Returns

the pointer to the note editor's backend

4.49.2.2 clear()

```
void quentier::NoteEditor::clear ( )
```

Clear the contents of the note editor

4.49.2.3 convertToNote

```
void quentier::NoteEditor::convertToNote ( )  [slot]
```

Invoke this slot to launch the asynchronous procedure of converting the current contents of the note editor to note; the convertedToNote signal would be emitted in response when the conversion is done

4.49.2.4 currentNoteLocalUid()

```
QString quentier::NoteEditor::currentNoteLocalUid ( ) const
```

Get the local uid of the note currently set to the note editor

4.49.2.5 defaultFont()

```
const QFont* quentier::NoteEditor::defaultFont ( ) const
```

Returns

pointer to the default font used by the note editor; if no such font was set to the editor previously, returns null pointer

4.49.2.6 defaultPalette()

```
QPalette quentier::NoteEditor::defaultPalette ( ) const
```

Returns

palette containing default colors used by the editor; the palette is composed of colors from note editor widget's native palette but some of them might be overridden by colors from the palette specified previously via set↩ DefaultPalette method: those colors from the specified palette which were valid

4.49.2.7 idleTime()

```
qint64 quentier::NoteEditor::idleTime ( ) const
```

Returns

the number of milliseconds since the last user's interaction with the note editor or -1 if there was no interaction or if no note is loaded at the moment

4.49.2.8 inAppNoteLinkPasteRequested

```
void quentier::NoteEditor::inAppNoteLinkPasteRequested (
            QString url,
            QString userId,
            QString shardId,
            QString noteGuid )  [signal]
```

inAppNoteLinkPasteRequested signal is emitted when the note editor detects the attempt to paste the in-app note link into the note editor; the link would not be inserted right away, instead this signal would be emitted. Whatever party managing the note editor is expected to connect some slot to this signal and provide the optionally amended link information to the note editor by sending the signal connected to its insertInAppNoteLink slot - this slot accepts both the URL of the link and the link text and performs the actual link insertion into the note. If the link text is empty, the URL itself is used as the link text.

4.49.2.9 initialize()

```
void quentier::NoteEditor::initialize (
            LocalStorageManagerAsync & localStorageManager,
            SpellChecker & spellChecker,
            const Account & account,
            QThread * pBackgroundJobsThread = nullptr )
```

NoteEditor requires LocalStorageManagerAsync, SpellChecker and Account for its work but due to the particularities of Qt's .ui files processing these can't be passed right inside the constructor, hence here's a special initialization method

Parameters

localStorageManager	The reference to LocalStorageManagerAsync, to set up signal-slot connections with it
spellChecker	The spell checker to be used by note editor for, well, spell-checking
account	Currently active account
pBackgroundJobsThread	Pointer to the thread to be used for scheduling of background jobs of NoteEditor; if null, NoteEditor's background jobs would take place in GUI thread

4.49.2.10 isEditorPageModified()

```
bool quentier::NoteEditor::isEditorPageModified ( ) const
```

Returns

true if there's content within the editor not yet converted to note, false otherwise

4.49.2.11 isModified()

```
bool quentier::NoteEditor::isModified ( ) const
```

Returns

true if there's content within the editor not yet converted to note or not saved to local storage, false otherwise

4.49.2.12 isNoteLoaded()

```
bool quentier::NoteEditor::isNoteLoaded ( ) const
```

Returns

true if the note last set to the editor has been fully loaded already, false otherwise

4.49.2.13 saveNoteToLocalStorage

```
void quentier::NoteEditor::saveNoteToLocalStorage ( )  [slot]
```

Invoke this slot to launch the asynchronous procedure of saving the modified current note back to the local storage. If no note is set to the editor or if the note is not modified, no action would be performed. Otherwise noteSaved↩ ToLocalStorage signal would be emitted in case of successful saving or failedToSaveNoteToLocalStorage would be emitted otherwise

4.49.2.14 setAccount()

```
void quentier::NoteEditor::setAccount (
            const Account & account )
```

Set the current account to the note editor

4.49.2.15 setBackend()

```
void quentier::NoteEditor::setBackend (
            INoteEditorBackend * backend )
```

This method can be used to set the backend to the note editor; the note editor has the default backend so this method is not obligatory to be called

4.49.2.16 setCurrentNoteLocalUid()

```
void quentier::NoteEditor::setCurrentNoteLocalUid (
            const QString & noteLocalUid )
```

Set note local uid to the note editor. The note is being searched for within the local storage, in case of no note being found noteNotFound signal is emitted. Otherwise note editor page starts loading.

Parameters

noteLocalUid	The local uid of note

4.49.2.17 setDefaultFont

```
void quentier::NoteEditor::setDefaultFont (
            const QFont & font )  [slot]
```

Sets the font which would be used by the editor by default

Parameters

font	The font to be used by the editor by default

4.49.2.18 setDefaultPalette

```
void quentier::NoteEditor::setDefaultPalette (
            const QPalette & pal )  [slot]
```

Sets the palette with colors to be used by the editor. New colors are applied after the note is fully loaded. If no note is set to the editor, the palette is simply remembered for the next note to be loaded into it.

Colors within the palette and their usage:

1. WindowText - used as default font color
2. Base - used as default background color
3. HighlightedText - used as font color for selected text
4. Highlight - used as background color for selected text

Parameters

pal	The palette to be set. Invalid colors from it are substituted by colors from widget's palette by the editor

4.49.2.19 setFocus()

```
void quentier::NoteEditor::setFocus ( )
```

Sets the focus to the backend note editor widget

4.49.2.20 setInitialPageHtml()

```
void quentier::NoteEditor::setInitialPageHtml (
            const QString & html )
```

Set the html to be displayed when the note is not set to the editor

4.49.2.21 setNoteDeletedPageHtml()

```
void quentier::NoteEditor::setNoteDeletedPageHtml (
            const QString & html )
```

Set the html to be displayed when the note set to the editor was deleted from the local storage (either marked as deleted or deleted permanently i.e. expunged)

4.49.2.22 setNoteLoadingPageHtml()

```
void quentier::NoteEditor::setNoteLoadingPageHtml (
            const QString & html )
```

Set the html to be displayed when the note set to the editor is being loaded into it

4.49.2.23 setNoteNotFoundPageHtml()

```
void quentier::NoteEditor::setNoteNotFoundPageHtml (
            const QString & html )
```

Set the html to be displayed when the note attempted to be set to the editor was not found within the local storage

4.49.2.24 setNoteTitle

```
void quentier::NoteEditor::setNoteTitle (
            const QString & noteTitle )  [slot]
```

Invoke this slot to set the title to the note displayed via the note editor. The note editor itself doesn't manage the note title in any way so any external code using the note editor can set the title to the note editor's note which would be considered modified if the title is new and then eventually the note would be saved to local storage

Parameters

noteTitle	The title of the note

4.49.2.25 setTagIds

```
void quentier::NoteEditor::setTagIds (
```

```
            const QStringList & tagLocalUids,
            const QStringList & tagGuids )  [slot]
```

Invoke this slot to set tag local uids and/or tag guids to the note displayed via the note editor. The note editor itself doesn't manage the note tags in any way so any external code using the note editor can set the tag ids to the note editor's internal note which would be considered modified if the tag ids are new and then eventually the note would be saved to local storage

Parameters

tagLocalUids	The list of tag local uids for the note
tagGuids	The list of tag guids for the note

4.49.2.26 setUndoStack()

```
void quentier::NoteEditor::setUndoStack (
            QUndoStack * pUndoStack )
```

Set the undo stack for the note editor to use

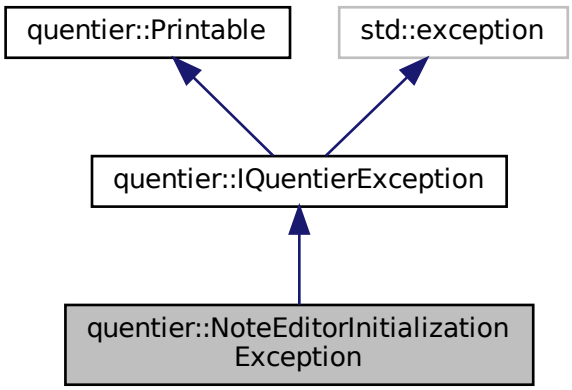
4.49.2.27 undoStack()

```
const QUndoStack* quentier::NoteEditor::undoStack ( ) const
```

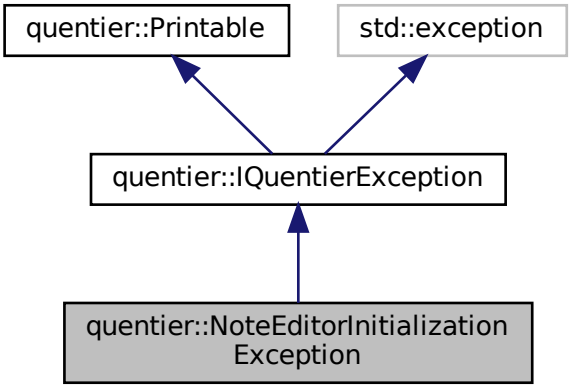
Get the undo stack serving to the note editor

4.50 quentier::NoteEditorInitializationException Class Reference

Inheritance diagram for quentier::NoteEditorInitializationException:



Collaboration diagram for quentier::NoteEditorInitializationException:



Public Member Functions

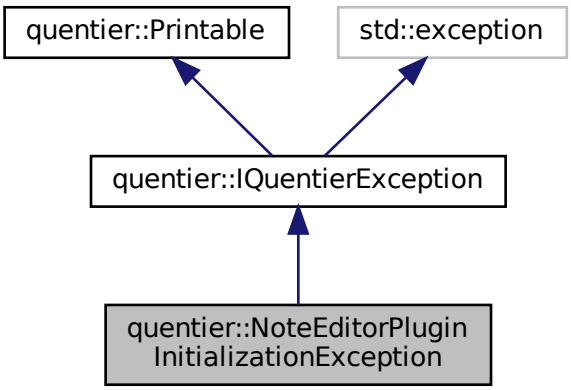
- **NoteEditorInitializationException** (const ErrorString &message)

Protected Member Functions

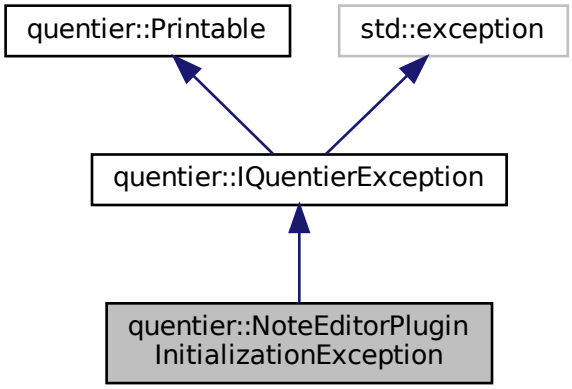
- virtual const QString **exceptionDisplayName** () const override

4.51 quentier::NoteEditorPluginInitializationException Class Reference

Inheritance diagram for quentier::NoteEditorPluginInitializationException:



Collaboration diagram for quentier::NoteEditorPluginInitializationException:



Public Member Functions

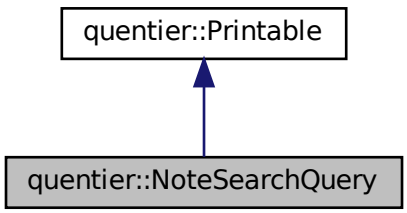
- **NoteEditorPluginInitializationException** (const ErrorString &message)

Protected Member Functions

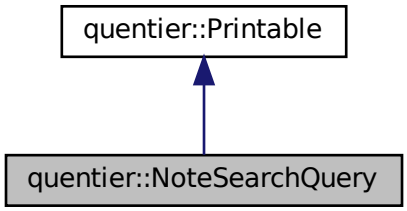
- virtual const QString **exceptionDisplayName** () const override

4.52 quentier::NoteSearchQuery Class Reference

Inheritance diagram for quentier::NoteSearchQuery:



Collaboration diagram for quentier::NoteSearchQuery:



Public Member Functions

- **NoteSearchQuery** (const NoteSearchQuery &other)
- **NoteSearchQuery** (NoteSearchQuery &&other)
- NoteSearchQuery & **operator=** (const NoteSearchQuery &other)
- NoteSearchQuery & **operator=** (NoteSearchQuery &&other)
- bool **isEmpty** () const
- void **clear** ()
- const QString queryString () const
- bool **setQueryString** (const QString &queryString, ErrorString &error)
- const QString notebookModifier () const
- bool **hasAnyModifier** () const
- const QStringList & **tagNames** () const
- const QStringList & **negatedTagNames** () const
- bool **hasAnyTag** () const
- bool **hasNegatedAnyTag** () const
- const QStringList & **titleNames** () const
- const QStringList & **negatedTitleNames** () const
- bool **hasAnyTitleName** () const
- bool **hasNegatedAnyTitleName** () const
- const QVector< qint64 > & **creationTimestamps** () const
- const QVector< qint64 > & **negatedCreationTimestamps** () const
- bool **hasAnyCreationTimestamp** () const
- bool **hasNegatedAnyCreationTimestamp** () const
- const QVector< qint64 > & **modificationTimestamps** () const
- const QVector< qint64 > & **negatedModificationTimestamps** () const
- bool **hasAnyModificationTimestamp** () const
- bool **hasNegatedAnyModificationTimestamp** () const
- const QStringList & **resourceMimeTypes** () const
- const QStringList & **negatedResourceMimeTypes** () const
- bool **hasAnyResourceMimeType** () const
- bool **hasNegatedAnyResourceMimeType** () const
- const QVector< qint64 > & **subjectDateTimestamps** () const
- const QVector< qint64 > & **negatedSubjectDateTimestamps** () const
- bool **hasAnySubjectDateTimestamp** () const
- bool **hasNegatedAnySubjectDateTimestamp** () const
- const QVector< double > & **latitudes** () const

- const QVector< double > & **negatedLatitudes** () const
- bool **hasAnyLatitude** () const
- bool **hasNegatedAnyLatitude** () const
- const QVector< double > & **longitudes** () const
- const QVector< double > & **negatedLongitudes** () const
- bool **hasAnyLongitude** () const
- bool **hasNegatedAnyLongitude** () const
- const QVector< double > & **altitudes** () const
- const QVector< double > & **negatedAltitudes** () const
- bool **hasAnyAltitude** () const
- bool **hasNegatedAnyAltitude** () const
- const QStringList & **authors** () const
- const QStringList & **negatedAuthors** () const
- bool **hasAnyAuthor** () const
- bool **hasNegatedAnyAuthor** () const
- const QStringList & **sources** () const
- const QStringList & **negatedSources** () const
- bool **hasAnySource** () const
- bool **hasNegatedAnySource** () const
- const QStringList & **sourceApplications** () const
- const QStringList & **negatedSourceApplications** () const
- bool **hasAnySourceApplication** () const
- bool **hasNegatedAnySourceApplication** () const
- const QStringList & **contentClasses** () const
- const QStringList & **negatedContentClasses** () const
- bool **hasAnyContentClass** () const
- bool **hasNegatedAnyContentClass** () const
- const QStringList & **placeNames** () const
- const QStringList & **negatedPlaceNames** () const
- bool **hasAnyPlaceName** () const
- bool **hasNegatedAnyPlaceName** () const
- const QStringList & **applicationData** () const
- const QStringList & **negatedApplicationData** () const
- bool **hasAnyApplicationData** () const
- bool **hasNegatedAnyApplicationData** () const
- const QVector< qint64 > & **reminderOrders** () const
- const QVector< qint64 > & **negatedReminderOrders** () const
- bool **hasAnyReminderOrder** () const
- bool **hasNegatedAnyReminderOrder** () const
- const QVector< qint64 > & **reminderTimes** () const
- const QVector< qint64 > & **negatedReminderTimes** () const
- bool **hasAnyReminderTime** () const
- bool **hasNegatedAnyReminderTime** () const
- const QVector< qint64 > & **reminderDoneTimes** () const
- const QVector< qint64 > & **negatedReminderDoneTimes** () const
- bool **hasAnyReminderDoneTime** () const
- bool **hasNegatedAnyReminderDoneTime** () const
- bool **hasUnfinishedToDo** () const
- bool **hasNegatedUnfinishedToDo** () const
- bool **hasFinishedToDo** () const
- bool **hasNegatedFinishedToDo** () const
- bool **hasAnyToDo** () const
- bool **hasNegatedAnyToDo** () const
- bool **hasEncryption** () const
- bool **hasNegatedEncryption** () const

- const QStringList & **contentSearchTerms** () const
- const QStringList & **negatedContentSearchTerms** () const
- bool **hasAnyContentSearchTerms** () const
- bool **isMatcheable** () const
- virtual QTextStream & **print** (QTextStream &strm) const override

Additional Inherited Members

4.52.1 Member Function Documentation

4.52.1.1 notebookModifier()

```
const QString quentier::NoteSearchQuery::notebookModifier ( ) const
```

If query string has "notebook:<notebook name>" scope modifier, this method returns the name of the notebook, otherwise it returns empty string

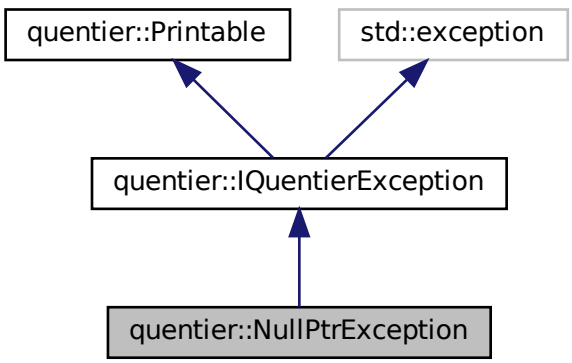
4.52.1.2 queryString()

```
const QString quentier::NoteSearchQuery::queryString ( ) const
```

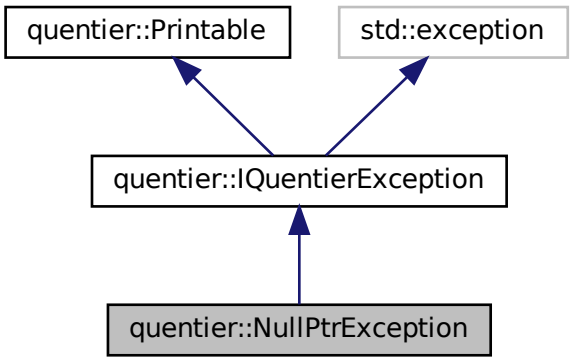
Returns the original non-parsed query string

4.53 quentier::NullPtrException Class Reference

Inheritance diagram for quentier::NullPtrException:



Collaboration diagram for quentier::NullPtrException:



Public Member Functions

- **NullPtrException** (const ErrorString &message)

Protected Member Functions

- virtual const QString **exceptionDisplayName** () const override

4.54 quentier::ResourceRecognitionIndexItem::ObjectItem Struct Reference

Public Member Functions

- bool **operator==** (const ObjectItem &other) const

Public Attributes

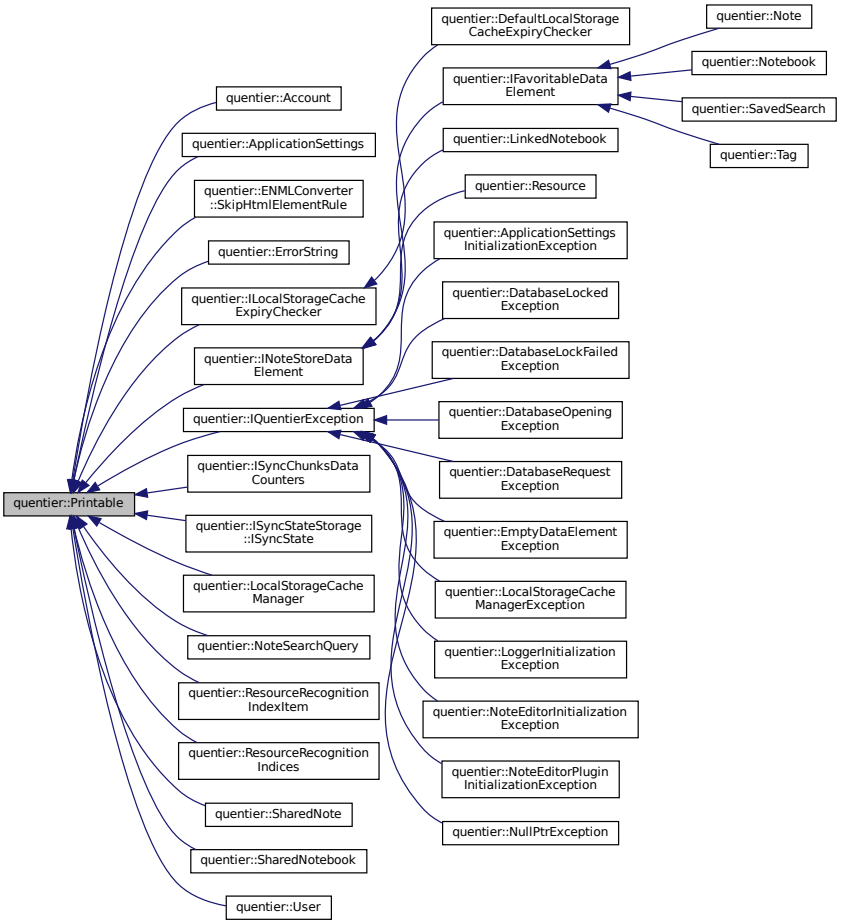
- QString **m_objectType**
- int **m_weight** = -1

4.55 quentier::Printable Class Reference

The Printable class is the interface for Quentier's internal classes which should be able to write themselves into QTextStream and/or convert to QString.

`#include <Printable.h>`

Inheritance diagram for quentier::Printable:



Public Member Functions

- virtual QTextStream & **print** (QTextStream &strm) const =0
- virtual const QString **toString** () const

Protected Member Functions

- **Printable** (const Printable &other)
- Printable & **operator=** (const Printable &other)

Friends

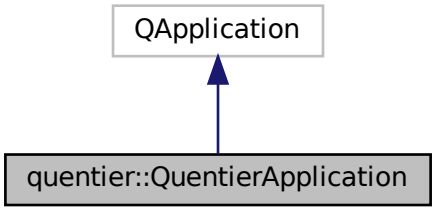
- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const Printable &printable)
- QUENTIER_EXPORT QDebug & **operator**<< (QDebug &debug, const Printable &printable)

4.55.1 Detailed Description

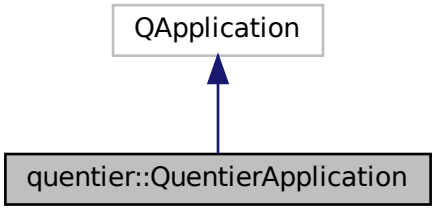
The Printable class is the interface for Quentier's internal classes which should be able to write themselves into QTextStream and/or convert to QString.

4.56 quentier::QuentierApplication Class Reference

Inheritance diagram for quentier::QuentierApplication:



Collaboration diagram for quentier::QuentierApplication:



Public Member Functions

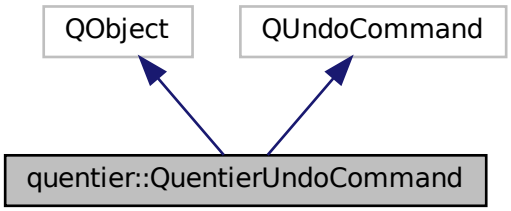
- **QuentierApplication** (int &argc, char *argv[])
- virtual bool **notify** (QObject *pObject, QEvent *pEvent) override
- virtual bool **event** (QEvent *pEvent) override

4.57 quentier::QuentierUndoCommand Class Reference

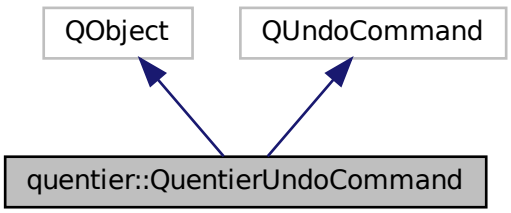
The QuentierUndoCommand class has the sole purpose of working around one quirky aspect of Qt's undo/redo framework: when you push QUndoCommand to QUndoStack, it calls "redo" method of that command. This class offers subclasses to implement their own methods for actual "undo" and "redo" commands while ignoring the attempts to "redo" anything if there were no previous "undo" call prior to that.

`#include <QuentierUndoCommand.h>`

Inheritance diagram for quentier::QuentierUndoCommand:



Collaboration diagram for quentier::QuentierUndoCommand:



Signals

- void **notifyError** (ErrorString error)

Public Member Functions

- **QuentierUndoCommand** (QUndoCommand *parent=nullptr)
- **QuentierUndoCommand** (const QString &text, QUndoCommand *parent=nullptr)
- virtual void **undo** () override final
- virtual void **redo** () override final
- bool **onceUndoExecuted** () const

Protected Member Functions

- virtual void **undoImpl** ()=0
- virtual void **redoImpl** ()=0

4.57.1 Detailed Description

The QuentierUndoCommand class has the sole purpose of working around one quirky aspect of Qt's undo/redo framework: when you push QUndoCommand to QUndoStack, it calls "redo" method of that command. This class offers subclasses to implement their own methods for actual "undo" and "redo" commands while ignoring the attempts to "redo" anything if there were no previous "undo" call prior to that.

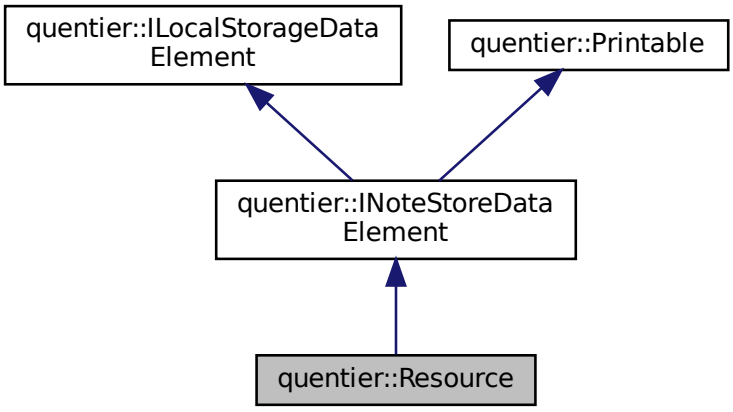
The rationale behind the current behaviour seems to be the compliance with "command pattern behaviour" when you create the command to execute the action instead of just executing it immediately. This design is enforced by Qt's undo/redo framework, there's no option to choose not to call "redo" when pushing to the stack.

One thing which this design fails to see is the fact that the command may be already executed externally by the moment the QUndoCommand can be created. Suppose we can get the information about how to undo (and then again redo) that command. We create the corresponding QUndoCommand, set up the stuff for its undo/redo methods and push it to QUndoStack for future use... But at the same time QUndoStack calls "redo" method of the command. Really not the behaviour you'd like to have.

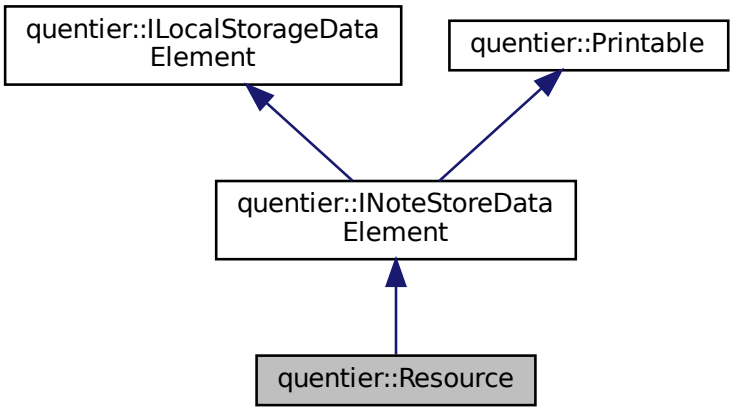
QuentierUndoCommand is also QObject, it is for error reporting via notifyError signal

4.58 quentier::Resource Class Reference

Inheritance diagram for quentier::Resource:



Collaboration diagram for quentier::Resource:



Public Member Functions

- **Resource** (const Resource &other)
- **Resource** (Resource &&other)
- **Resource** (const qevercloud::Resource &resource)
- Resource & **operator=** (const Resource &other)
- Resource & **operator=** (Resource &&other)
- bool **operator==** (const Resource &other) const
- bool **operator!=** (const Resource &other) const
- const qevercloud::Resource & **qevercloudResource** () const
- qevercloud::Resource & **qevercloudResource** ()
- virtual void **clear** () override
- virtual bool **hasGuid** () const override
- virtual const QString & **guid** () const override
- virtual void **setGuid** (const QString &guid) override
- virtual bool **hasUpdateSequenceNumber** () const override
- virtual qint32 **updateSequenceNumber** () const override
- virtual void **setUpdateSequenceNumber** (const qint32 updateSequenceNumber) override
- virtual bool **checkParameters** (ErrorString &errorDescription) const override
- QString **displayName** () const
- void **setDisplayName** (const QString &displayName)
- QString **preferredFileSuffix** () const
- int **indexInNote** () const
- void **setIndexInNote** (const int index)
- bool **hasNoteGuid** () const
- const QString & **noteGuid** () const
- void **setNoteGuid** (const QString &guid)
- bool **hasNoteLocalUid** () const
- const QString & **noteLocalUid** () const
- void **setNoteLocalUid** (const QString &guid)
- bool **hasData** () const
- bool **hasDataHash** () const

- const QByteArray & **dataHash** () const
- void **setDataHash** (const QByteArray &hash)
- bool **hasDataSize** () const
- qint32 **dataSize** () const
- void **setDataSize** (const qint32 size)
- bool **hasDataBody** () const
- const QByteArray & **dataBody** () const
- void **setDataBody** (const QByteArray &body)
- bool **hasMime** () const
- const QString & **mime** () const
- void **setMime** (const QString &mime)
- bool **hasWidth** () const
- qint16 **width** () const
- void **setWidth** (const qint16 width)
- bool **hasHeight** () const
- qint16 **height** () const
- void **setHeight** (const qint16 height)
- bool **hasRecognitionData** () const
- bool **hasRecognitionDataHash** () const
- const QByteArray & **recognitionDataHash** () const
- void **setRecognitionDataHash** (const QByteArray &hash)
- bool **hasRecognitionDataSize** () const
- qint32 **recognitionDataSize** () const
- void **setRecognitionDataSize** (const qint32 size)
- bool **hasRecognitionDataBody** () const
- const QByteArray & **recognitionDataBody** () const
- void **setRecognitionDataBody** (const QByteArray &body)
- bool **hasAlternateData** () const
- bool **hasAlternateDataHash** () const
- const QByteArray & **alternateDataHash** () const
- void **setAlternateDataHash** (const QByteArray &hash)
- bool **hasAlternateDataSize** () const
- qint32 **alternateDataSize** () const
- void **setAlternateDataSize** (const qint32 size)
- bool **hasAlternateDataBody** () const
- const QByteArray & **alternateDataBody** () const
- void **setAlternateDataBody** (const QByteArray &body)
- bool **hasResourceAttributes** () const
- const qevercloud::ResourceAttributes & **resourceAttributes** () const
- qevercloud::ResourceAttributes & **resourceAttributes** ()
- void **setResourceAttributes** (const qevercloud::ResourceAttributes &attributes)
- void **setResourceAttributes** (qevercloud::ResourceAttributes &&attributes)
- virtual QTextStream & **print** (QTextStream &strm) const override

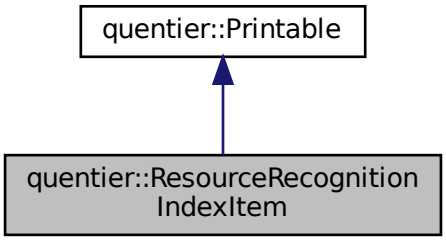
Friends

- class **Note**

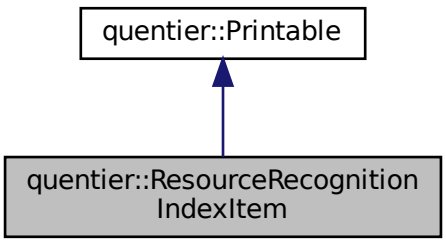
Additional Inherited Members

4.59 quentier::ResourceRecognitionIndexItem Class Reference

Inheritance diagram for quentier::ResourceRecognitionIndexItem:



Collaboration diagram for quentier::ResourceRecognitionIndexItem:



Classes

- struct BarcodeItem
- struct ObjectItem
- struct ShapeItem
- struct TextItem

Public Member Functions

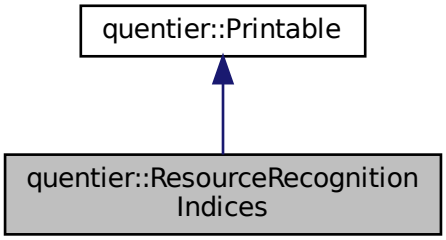
- **ResourceRecognitionIndexItem** (const ResourceRecognitionIndexItem &other)
- ResourceRecognitionIndexItem & **operator=** (const ResourceRecognitionIndexItem &other)
- bool **isValid** () const

- int **x** () const
- void **setX** (const int x)
- int **y** () const
- void **setY** (const int y)
- int **h** () const
- void **setH** (const int h)
- int **w** () const
- void **setW** (const int w)
- int **offset** () const
- void **setOffset** (const int offset)
- int **duration** () const
- void **setDuration** (const int duration)
- QVector< int > **strokeList** () const
- int **numStrokes** () const
- bool **strokeAt** (const int strokeIndex, int &stroke) const
- bool **setStrokeAt** (const int strokeIndex, const int stroke)
- void **setStrokeList** (const QVector< int > &strokeList)
- void **reserveStrokeListSpace** (const int numItems)
- void **addStroke** (const int stroke)
- bool **removeStroke** (const int stroke)
- bool **removeStrokeAt** (const int strokeIndex)
- QVector< TextItem > **textItems** () const
- int **numTextItems** () const
- bool **textItemAt** (const int textItemIndex, TextItem &textItem) const
- bool **setTextItemAt** (const int textItemIndex, const TextItem &textItem)
- void **setTextItems** (const QVector< TextItem > &textItems)
- void **reserveTextItemsSpace** (const int numItems)
- void **addTextItem** (const TextItem &item)
- bool **removeTextItem** (const TextItem &item)
- bool **removeTextItemAt** (const int textItemIndex)
- QVector< ObjectItem > **objectItems** () const
- int **numObjectItems** () const
- bool **objectItemAt** (const int objectItemIndex, ObjectItem &objectItem) const
- bool **setObjectItemAt** (const int objectItemIndex, const ObjectItem &objectItem)
- void **setObjectItems** (const QVector< ObjectItem > &objectItems)
- void **reserveObjectItemsSpace** (const int numItems)
- void **addObjectItem** (const ObjectItem &item)
- bool **removeObjectItem** (const ObjectItem &item)
- bool **removeObjectItemAt** (const int objectItemIndex)
- QVector< ShapeItem > **shapeItems** () const
- int **numShapeItems** () const
- bool **shapeItemAt** (const int shapeItemIndex, ShapeItem &shapeItem) const
- bool **setShapeItemAt** (const int shapeItemIndex, const ShapeItem &shapeItem)
- void **setShapeItems** (const QVector< ShapeItem > &shapeItems)
- void **reserveShapeItemsSpace** (const int numItems)
- void **addShapeItem** (const ShapeItem &item)
- bool **removeShapeItem** (const ShapeItem &item)
- bool **removeShapeItemAt** (const int shapeItemIndex)
- QVector< BarcodeItem > **barcodeItems** () const
- int **numBarcodeItems** () const
- bool **barcodeItemAt** (const int barcodeItemIndex, BarcodeItem &barcodeItem) const
- bool **setBarcodeItemAt** (const int barcodeItemIndex, const BarcodeItem &barcodeItem)
- void **setBarcodeItems** (const QVector< BarcodeItem > &barcodeItems)
- void **reserveBarcodeItemsSpace** (const int numItems)
- void **addBarcodeItem** (const BarcodeItem &item)
- bool **removeBarcodeItem** (const BarcodeItem &item)
- bool **removeBarcodeItemAt** (const int barcodeItemIndex)
- virtual QTextStream & **print** (QTextStream &strm) const override

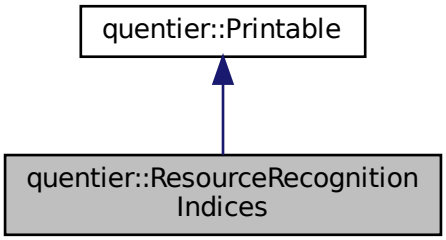
Additional Inherited Members

4.60 quentier::ResourceRecognitionIndices Class Reference

Inheritance diagram for quentier::ResourceRecognitionIndices:



Collaboration diagram for quentier::ResourceRecognitionIndices:



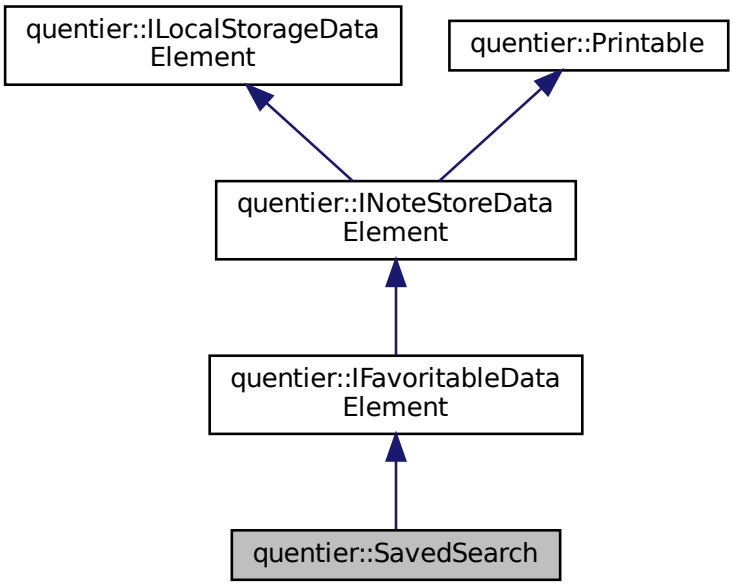
Public Member Functions

- **ResourceRecognitionIndices** (const ResourceRecognitionIndices &other)
- **ResourceRecognitionIndices** (const QByteArray &rawRecognitionIndicesData)
- ResourceRecognitionIndices & **operator=** (const ResourceRecognitionIndices &other)
- bool **isNull** () const
- bool **isValid** () const
- QString **objectId** () const
- QString **objectType** () const
- QString **recoType** () const
- QString **engineVersion** () const
- QString **docType** () const
- QString **lang** () const
- int **objectHeight** () const
- int **objectWidth** () const
- QVector< ResourceRecognitionIndexItem > **items** () const
- bool **setData** (const QByteArray &rawRecognitionIndicesData)
- virtual QTextStream & **print** (QTextStream &strm) const override

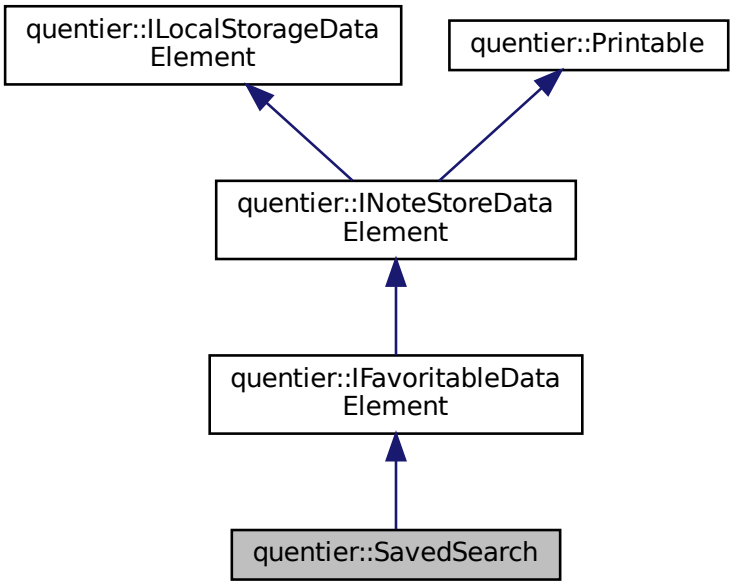
Additional Inherited Members

4.61 quentier::SavedSearch Class Reference

Inheritance diagram for quentier::SavedSearch:



Collaboration diagram for quentier::SavedSearch:



Public Types

- using **QueryFormat** = qevercloud::QueryFormat
- using **SavedSearchScope** = qevercloud::SavedSearchScope

Public Member Functions

- **SavedSearch** (const SavedSearch &other)
- **SavedSearch** (SavedSearch &&other)
- SavedSearch & **operator=** (const SavedSearch &other)
- SavedSearch & **operator=** (SavedSearch &&other)
- **SavedSearch** (const qevercloud::SavedSearch &search)
- **SavedSearch** (qevercloud::SavedSearch &&search)
- const qevercloud::SavedSearch & **qevercloudSavedSearch** () const
- qevercloud::SavedSearch & **qevercloudSavedSearch** ()
- bool **operator==** (const SavedSearch &other) const
- bool **operator!=** (const SavedSearch &other) const
- virtual void **clear** () override
- virtual bool **hasGuid** () const override
- virtual const QString & **guid** () const override
- virtual void **setGuid** (const QString &guid) override
- virtual bool **hasUpdateSequenceNumber** () const override
- virtual qint32 **updateSequenceNumber** () const override
- virtual void **setUpdateSequenceNumber** (const qint32 usn) override
- virtual bool **checkParameters** (ErrorString &errorDescription) const override
- bool **hasName** () const

- const QString & **name** () const
- void **setName** (const QString &name)
- bool **hasQuery** () const
- const QString & **query** () const
- void **setQuery** (const QString &query)
- bool **hasQueryFormat** () const
- QueryFormat **queryFormat** () const
- void **setQueryFormat** (const qint8 queryFormat)
- bool **hasIncludeAccount** () const
- bool **includeAccount** () const
- void **setIncludeAccount** (const bool includeAccount)
- bool **hasIncludePersonalLinkedNotebooks** () const
- bool **includePersonalLinkedNotebooks** () const
- void **setIncludePersonalLinkedNotebooks** (const bool includePersonalLinkedNotebooks)
- bool **hasIncludeBusinessLinkedNotebooks** () const
- bool **includeBusinessLinkedNotebooks** () const
- void **setIncludeBusinessLinkedNotebooks** (const bool includeBusinessLinkedNotebooks)
- virtual QTextStream & **print** (QTextStream &strm) const override

Static Public Member Functions

- static bool **validateName** (const QString &name, ErrorString *pErrorDescription=nullptr)

Additional Inherited Members

4.62 quentier::ResourceRecognitionIndexItem::ShapeItem Struct Reference

Public Member Functions

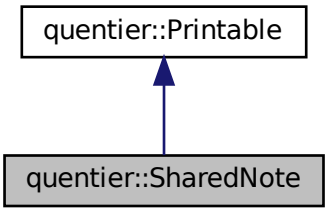
- bool **operator==** (const ShapeItem &other) const

Public Attributes

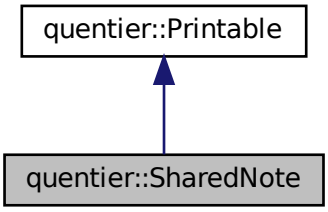
- QString **m_shapeType**
- int **m_weight** = -1

4.63 quentier::SharedNote Class Reference

Inheritance diagram for quentier::SharedNote:



Collaboration diagram for quentier::SharedNote:



Public Types

- using **SharedNotePrivilegeLevel** = qevercloud::SharedNotePrivilegeLevel
- using **ContactType** = qevercloud::ContactType

Public Member Functions

- **SharedNote** (const SharedNote &other)
- **SharedNote** (SharedNote &&other)
- **SharedNote** (const qevercloud::SharedNote &sharedNote)
- SharedNote & **operator=** (const SharedNote &other)
- SharedNote & **operator=** (SharedNote &&other)
- bool **operator==** (const SharedNote &other) const
- bool **operator!=** (const SharedNote &other) const
- const qevercloud::SharedNote & **qevercloudSharedNote** () const
- qevercloud::SharedNote & **qevercloudSharedNote** ()
- const QString & **noteGuid** () const

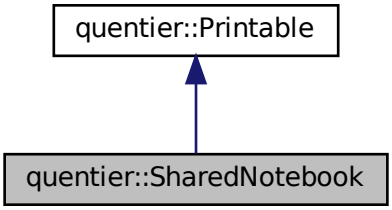
- void **setNoteGuid** (const QString &noteGuid)
- int **indexInNote** () const
- void **setIndexInNote** (const int index)
- bool **hasSharerUserId** () const
- qint32 **sharerUserId** () const
- void **setSharerUserId** (const qint32 id)
- bool **hasRecipientIdentityId** () const
- qint64 **recipientIdentityId** () const
- void **setRecipientIdentityId** (const qint64 identityId)
- bool **hasRecipientIdentityContactName** () const
- const QString & **recipientIdentityContactName** () const
- void **setRecipientIdentityContactName** (const QString &recipientIdentityContactName)
- bool **hasRecipientIdentityContactId** () const
- const QString & **recipientIdentityContactId** () const
- void **setRecipientIdentityContactId** (const QString &recipientIdentityContactId)
- bool **hasRecipientIdentityContactType** () const
- ContactType **recipientIdentityContactType** () const
- void **setRecipientIdentityContactType** (const ContactType recipientIdentityContactType)
- void **setRecipientIdentityContactType** (const qint32 recipientIdentityContactType)
- bool **hasRecipientIdentityContactPhotoUrl** () const
- const QString & **recipientIdentityContactPhotoUrl** () const
- void **setRecipientIdentityContactPhotoUrl** (const QString &recipientIdentityPhotoUrl)
- bool **hasRecipientIdentityContactPhotoLastUpdated** () const
- qint64 **recipientIdentityContactPhotoLastUpdated** () const
- void **setRecipientIdentityContactPhotoLastUpdated** (const qint64 recipientIdentityPhotoLastUpdated)
- bool **hasRecipientIdentityContactMessagingPermit** () const
- const QByteArray & **recipientIdentityContactMessagingPermit** () const
- void **setRecipientIdentityContactMessagingPermit** (const QByteArray &messagingPermit)
- bool **hasRecipientIdentityContactMessagingPermitExpires** () const
- qint64 **recipientIdentityContactMessagingPermitExpires** () const
- void **setRecipientIdentityContactMessagingPermitExpires** (const qint64 timestamp)
- bool **hasRecipientIdentityUserId** () const
- qint32 **recipientIdentityUserId** () const
- void **setRecipientIdentityUserId** (const qint32 userId)
- bool **hasRecipientIdentityDeactivated** () const
- bool **recipientIdentityDeactivated** () const
- void **setRecipientIdentityDeactivated** (const bool deactivated)
- bool **hasRecipientIdentitySameBusiness** () const
- bool **recipientIdentitySameBusiness** () const
- void **setRecipientIdentitySameBusiness** (const bool sameBusiness)
- bool **hasRecipientIdentityBlocked** () const
- bool **recipientIdentityBlocked** () const
- void **setRecipientIdentityBlocked** (const bool blocked)
- bool **hasRecipientIdentityUserConnected** () const
- bool **recipientIdentityUserConnected** () const
- void **setRecipientIdentityUserConnected** (const bool userConnected)
- bool **hasRecipientIdentityEventId** () const
- qint64 **recipientIdentityEventId** () const
- void **setRecipientIdentityEventId** (const qint64 eventId)
- bool **hasRecipientIdentity** () const
- const qevercloud::Identity & **recipientIdentity** () const
- void **setRecipientIdentity** (qevercloud::Identity &&identity)
- bool **hasPrivilegeLevel** () const
- SharedNotePrivilegeLevel **privilegeLevel** () const
- void **setPrivilegeLevel** (const SharedNotePrivilegeLevel level)

- void **setPrivilegeLevel** (const qint8 level)
- bool **hasCreationTimestamp** () const
- qint64 **creationTimestamp** () const
- void **setCreationTimestamp** (const qint64 timestamp)
- bool **hasModificationTimestamp** () const
- qint64 **modificationTimestamp** () const
- void **setModificationTimestamp** (const qint64 timestamp)
- bool **hasAssignmentTimestamp** () const
- qint64 **assignmentTimestamp** () const
- void **setAssignmentTimestamp** (const qint64 timestamp)
- virtual QTextStream & **print** (QTextStream &strm) const override

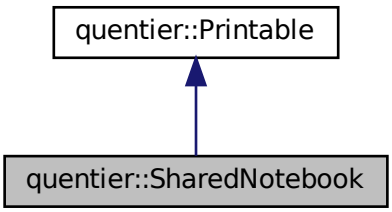
Additional Inherited Members

4.64 quentier::SharedNotebook Class Reference

Inheritance diagram for quentier::SharedNotebook:



Collaboration diagram for quentier::SharedNotebook:



Public Types

- using **SharedNotebookPrivilegeLevel** = qevercloud::SharedNotebookPrivilegeLevel

Public Member Functions

- **SharedNotebook** (const SharedNotebook &other)
- **SharedNotebook** (SharedNotebook &&other)
- **SharedNotebook** (const qevercloud::SharedNotebook &qecSharedNotebook)
- SharedNotebook & **operator=** (const SharedNotebook &other)
- SharedNotebook & **operator=** (SharedNotebook &&other)
- bool **operator==** (const SharedNotebook &other) const
- bool **operator!=** (const SharedNotebook &other) const
- const qevercloud::SharedNotebook & **qevercloudSharedNotebook** () const
- qevercloud::SharedNotebook & **qevercloudSharedNotebook** ()
- int **indexInNotebook** () const
- void **setIndexInNotebook** (const int index)
- bool **hasId** () const
- qint64 **id** () const
- void **setId** (const qint64 id)
- bool **hasUserId** () const
- qint32 **userId** () const
- void **setUserId** (const qint32 userId)
- bool **hasNotebookGuid** () const
- const QString & **notebookGuid** () const
- void **setNotebookGuid** (const QString &notebookGuid)
- bool **hasEmail** () const
- const QString & **email** () const
- void **setEmail** (const QString &email)
- bool **hasCreationTimestamp** () const
- qint64 **creationTimestamp** () const
- void **setCreationTimestamp** (const qint64 timestamp)
- bool **hasModificationTimestamp** () const
- qint64 **modificationTimestamp** () const
- void **setModificationTimestamp** (const qint64 timestamp)
- bool **hasUsername** () const
- const QString & **username** () const
- void **setUsername** (const QString &username)
- bool **hasPrivilegeLevel** () const
- SharedNotebookPrivilegeLevel **privilegeLevel** () const
- void **setPrivilegeLevel** (const SharedNotebookPrivilegeLevel privilegeLevel)
- void **setPrivilegeLevel** (const qint8 privilegeLevel)
- bool **hasReminderNotifyEmail** () const
- bool **reminderNotifyEmail** () const
- void **setReminderNotifyEmail** (const bool notifyEmail)
- bool **hasReminderNotifyApp** () const
- bool **reminderNotifyApp** () const
- void **setReminderNotifyApp** (const bool notifyApp)
- bool **hasRecipientUsername** () const
- const QString & **recipientUsername** () const
- void **setRecipientUsername** (const QString &recipientUsername)
- bool **hasRecipientUserId** () const
- qint32 **recipientUserId** () const
- void **setRecipientUserId** (const qint32 userId)
- bool **hasRecipientIdentityId** () const
- qint64 **recipientIdentityId** () const
- void **setRecipientIdentityId** (const qint64 recipientIdentityId)
- bool **hasGlobalId** () const
- const QString & **globalId** () const

- void **setGlobalId** (const QString &globalId)
- bool **hasSharerUserId** () const
- qint32 **sharerUserId** () const
- void **setSharerUserId** (qint32 sharerUserId)
- bool **hasAssignmentTimestamp** () const
- qint64 **assignmentTimestamp** () const
- void **setAssignmentTimestamp** (const qint64 timestamp)
- virtual QTextStream & **print** (QTextStream &strm) const override

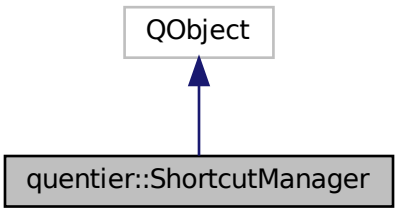
Friends

- class **Notebook**

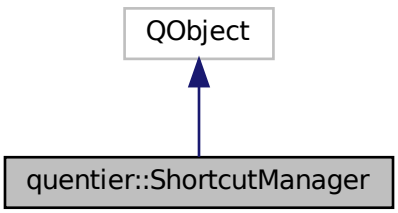
Additional Inherited Members

4.65 quentier::ShortcutManager Class Reference

Inheritance diagram for quentier::ShortcutManager:



Collaboration diagram for quentier::ShortcutManager:



Public Types

- enum **QuentierShortcutKey** {
  NewNote = 5000, **NewTag**, **NewNotebook**, **NewSavedSearch**,
  AddAttachment, **SaveAttachment**, **OpenAttachment**, **CopyAttachment**,
  CutAttachment, **RemoveAttachment**, **RenameAttachment**, **AddAccount**,
  ExitAccount, **SwitchAccount**, **AccountInfo**, **NoteSearch**,
  NewNoteSearch, **ShowNotes**, **ShowNotebooks**, **ShowTags**,
  ShowSavedSearches, **ShowDeletedNotes**, **ShowStatusBar**, **ShowToolBar**,
  PasteUnformatted, **Font**, **UpperIndex**, **LowerIndex**,
  AlignLeft, **AlignCenter**, **AlignRight**, **AlignFull**,
  IncreaseIndentation, **DecreaseIndentation**, **IncreaseFontSize**, **DecreaseFontSize**,
  InsertNumberedList, **InsertBulletedList**, **Strikethrough**, **Highlight**,
  InsertTable, **InsertRow**, **InsertColumn**, **RemoveRow**,
  RemoveColumn, **InsertHorizontalLine**, **InsertToDoTag**, **EditHyperlink**,
  CopyHyperlink, **RemoveHyperlink**, **Encrypt**, **Decrypt**,
  DecryptPermanently, **BackupLocalStorage**, **RestoreLocalStorage**, **UpgradeLocalStorage**,
  LocalStorageStatus, **SpellCheck**, **SpellCheckIgnoreWord**, **SpellCheckAddWordToUserDictionary**,
  SaveImage, **AnnotateImage**, **ImageRotateClockwise**, **ImageRotateCounterClockwise**,
  Synchronize, **FullSync**, **ImportFolders**, **Preferences**,
  ReleaseNotes, **ViewLogs**, **About**, **UnknownKey** = 100000 }

Public Slots

- void **setUserShortcut** (int key, QKeySequence shortcut, const Account &account, QString context={})
- void **setNonStandardUserShortcut** (QString nonStandardKey, QKeySequence shortcut, const Account &account, QString context={})
- void **setDefaultShortcut** (int key, QKeySequence shortcut, const Account &account, QString context={})
- void **setNonStandardDefaultShortcut** (QString nonStandardKey, QKeySequence shortcut, const Account &account, QString context={})

Signals

- void **shortcutChanged** (int key, QKeySequence shortcut, const Account &account, QString context)
- void **nonStandardShortcutChanged** (QString nonStandardKey, QKeySequence shortcut, const Account &account, QString context)

Public Member Functions

- **ShortcutManager** (QObject *parent=nullptr)
- QKeySequence shortcut (const int key, const Account &account, const QString &context={}) const
- QKeySequence shortcut (const QString &nonStandardKey, const Account &account, const QString &context={}) const
- QKeySequence defaultShortcut (const int key, const Account &account, const QString &context={}) const
- QKeySequence defaultShortcut (const QString &nonStandardKey, const Account &account, const QString &context={}) const
- QKeySequence userShortcut (const int key, const Account &account, const QString &context={}) const
- QKeySequence userShortcut (const QString &nonStandardKey, const Account &account, const QString &context={}) const

4.65.1 Member Function Documentation

4.65.1.1 defaultShortcut() [1/2]

```
QKeySequence quentier::ShortcutManager::defaultShortcut (
            const int key,
            const Account & account,
            const QString & context = {} ) const
```

Returns

Default shortcut for the standard key if present, otherwise empty key sequence

4.65.1.2 defaultShortcut() [2/2]

```
QKeySequence quentier::ShortcutManager::defaultShortcut (
            const QString & nonStandardKey,
            const Account & account,
            const QString & context = {} ) const
```

Returns

Default shortcut for the non-standard key if present, otherwise empty key sequence

4.65.1.3 shortcut() [1/2]

```
QKeySequence quentier::ShortcutManager::shortcut (
            const int key,
            const Account & account,
            const QString & context = {} ) const
```

Returns

Active shortcut for the standard key - either the user defined shortcut (if present) or the default one (if present as well)

4.65.1.4 shortcut() [2/2]

```
QKeySequence quentier::ShortcutManager::shortcut (
            const QString & nonStandardKey,
            const Account & account,
            const QString & context = {} ) const
```

Returns

Active shortcut for the non-standard key - either the user defined shortcut (if present) or the default one (if present as well)

4.65.1.5 userShortcut() [1/2]

```
QKeySequence quentier::ShortcutManager::userShortcut (
             const int key,
             const Account & account,
             const QString & context = {} ) const
```

Returns

User defined shortcut for the standard key if present, otherwise empty key sequence

4.65.1.6 userShortcut() [2/2]

```
QKeySequence quentier::ShortcutManager::userShortcut (
             const QString & nonStandardKey,
             const Account & account,
             const QString & context = {} ) const
```

Returns

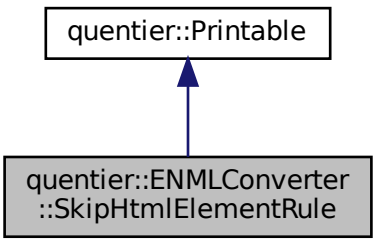
User defined shortcut for the non-standard key if present, otherwise empty key sequence

4.66 quentier::ENMLConverter::SkipHtmlElementRule Class Reference

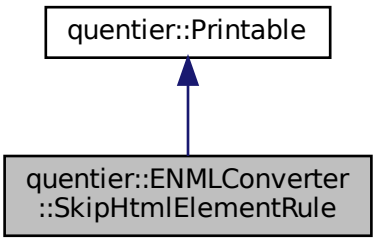
The SkipHtmlElementRule class describes the set of rules for HTML -> ENML conversion about the HTML elements that should not be actually converted to ENML due to their nature of being "helper" elements for the display or functioning of something within the note editor's page. The HTML -> ENML conversion would ignore tags and attributes forbidden by ENML even without these rules conditionally preserving or skipping the contents and nested elements of skipped elements.

```
#include <ENMLConverter.h>
```

Inheritance diagram for quentier::ENMLConverter::SkipHtmlElementRule:



Collaboration diagram for quentier::ENMLConverter::SkipHtmlElementRule:



Public Types

- enum **ComparisonRule** { **Equals** = 0, **StartsWith**, **EndsWith**, **Contains** }

Public Member Functions

- virtual QTextStream & **print** (QTextStream &strm) const override

Public Attributes

- QString **m_elementNameToSkip**
- ComparisonRule **m_elementNameComparisonRule** = ComparisonRule::Equals
- Qt::CaseSensitivity **m_elementNameCaseSensitivity** = Qt::CaseSensitive
- QString **m_attributeNameToSkip**
- ComparisonRule **m_attributeNameComparisonRule** = ComparisonRule::Equals
- Qt::CaseSensitivity **m_attributeNameCaseSensitivity** = Qt::CaseSensitive
- QString **m_attributeValueToSkip**
- ComparisonRule **m_attributeValueComparisonRule** = ComparisonRule::Equals
- Qt::CaseSensitivity **m_attributeValueCaseSensitivity** = Qt::CaseSensitive
- bool **m_includeElementContents** = false

Friends

- QUENTIER_EXPORT QTextStream & **operator**<< (QTextStream &strm, const ComparisonRule rule)

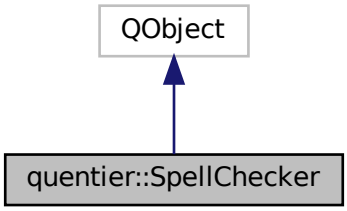
Additional Inherited Members

4.66.1 Detailed Description

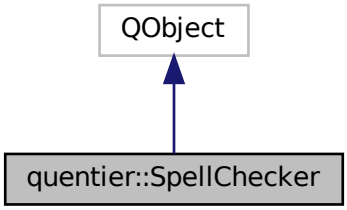
The SkipHtmlElementRule class describes the set of rules for HTML -> ENML conversion about the HTML elements that should not be actually converted to ENML due to their nature of being "helper" elements for the display or functioning of something within the note editor's page. The HTML -> ENML conversion would ignore tags and attributes forbidden by ENML even without these rules conditionally preserving or skipping the contents and nested elements of skipped elements.

4.67 quentier::SpellChecker Class Reference

Inheritance diagram for quentier::SpellChecker:



Collaboration diagram for quentier::SpellChecker:



Signals

- void **ready** ()

Public Member Functions

- **SpellChecker** (FileIOProcessorAsync *pFileIOProcessorAsync, const Account &account, QObject *parent=nullptr, const QString &userDictionaryPath={})
- QVector< std::pair< QString, bool > > **listAvailableDictionaries** () const
- void **setAccount** (const Account &account)
- void **enableDictionary** (const QString &language)
- void **disableDictionary** (const QString &language)
- bool **checkSpell** (const QString &word) const
- QStringList **spellCorrectionSuggestions** (const QString &misSpelledWord) const
- void **addToUserWordlist** (const QString &word)
- void **removeFromUserWordList** (const QString &word)
- void **ignoreWord** (const QString &word)
- void **removeWord** (const QString &word)
- bool **isReady** () const

4.68 quentier::StringUtils::StringFilterPredicate Struct Reference

Public Member Functions

- **StringFilterPredicate** (QSet< QString > &filteredStrings)
- bool **operator()** (const QString &str) const

Public Attributes

- QSet< QString > & **m_filteredStrings**

4.69 quentier::StringUtils Class Reference

Classes

- struct StringFilterPredicate

Public Member Functions

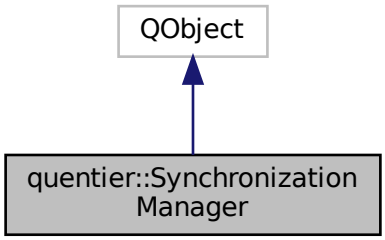
- void **removePunctuation** (QString &str, const QVector< QChar > &charactersToPreserve={}) const
- void **removeDiacritics** (QString &str) const
- void **removeNewlines** (QString &str) const

4.70 quentier::SynchronizationManager Class Reference

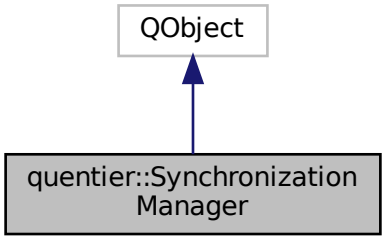
The SynchronizationManager class encapsulates methods and signals & slots required to perform the full or partial synchronization of data with remote Evernote servers. The class also deals with authentication with Evernote service through OAuth.

```
#include <SynchronizationManager.h>
```

Inheritance diagram for quentier::SynchronizationManager:



Collaboration diagram for quentier::SynchronizationManager:



Public Slots

- void setAccount (Account account)
- void authenticate ()
- void authenticateCurrentAccount ()
- void synchronize ()
- void stop ()
- void revokeAuthentication (const qevercloud::UserID userId)
- void setDownloadNoteThumbnails (bool flag)
- void setDownloadInkNoteImages (bool flag)
- void setInkNoteImagesStoragePath (QString path)

Signals

- void started ()
- void stopped ()
- void failed (ErrorString errorDescription)
- void finished (Account account, bool somethingDownloaded, bool somethingSent)
- void authenticationRevoked (bool success, ErrorString errorDescription, qevercloud::UserID userId)
- void authenticationFinished (bool success, ErrorString errorDescription, Account account)
- void remoteToLocalSyncStopped ()
- void sendLocalChangesStopped ()
- void willRepeatRemoteToLocalSyncAfterSendingChanges ()
- void detectedConflictDuringLocalChangesSending ()
- void rateLimitExceeded (qint32 secondsToWait)
- void remoteToLocalSyncDone (bool somethingDownloaded)
- void syncChunksDownloadProgress (qint32 highestDownloadedUsn, qint32 highestServerUsn, qint32 last↩ PreviousUsn)
- void syncChunksDownloaded ()
- void syncChunksDataProcessingProgress (ISyncChunksDataCountersPtr counters)
- void linkedNotebookSyncChunksDownloadProgress (qint32 highestDownloadedUsn, qint32 highestServer↩ Usn, qint32 lastPreviousUsn, LinkedNotebook linkedNotebook)
- void linkedNotebooksSyncChunksDownloaded ()
- void linkedNotebookSyncChunksDataProcessingProgress (ISyncChunksDataCountersPtr counters)
- void notesDownloadProgress (quint32 notesDownloaded, quint32 totalNotesToDownload)
- void linkedNotebooksNotesDownloadProgress (quint32 notesDownloaded, quint32 totalNotesToDownload)

- void resourcesDownloadProgress (quint32 resourcesDownloaded, quint32 totalResourcesToDownload)
- void linkedNotebooksResourcesDownloadProgress (quint32 resourcesDownloaded, quint32 totalResourcesToDownload)
- void preparedDirtyObjectsForSending ()
- void preparedLinkedNotebooksDirtyObjectsForSending ()
- void setAccountDone (Account account)
- void setDownloadNoteThumbnailsDone (bool flag)
- void setDownloadInkNoteImagesDone (bool flag)
- void setInkNoteImagesStoragePathDone (QString path)

Public Member Functions

- SynchronizationManager (QString host, LocalStorageManagerAsync &localStorageManagerAsync, IAuthenticationManager &authenticationManager, QObject *parent=nullptr, INoteStorePtr pNoteStore={}, IUserStorePtr pUserStore={}, IKeychainServicePtr pKeychainService={}, ISyncStateStoragePtr pSyncStateStorage={})
- bool active () const
- bool downloadNoteThumbnailsOption () const

4.70.1 Detailed Description

The SynchronizationManager class encapsulates methods and signals & slots required to perform the full or partial synchronization of data with remote Evernote servers. The class also deals with authentication with Evernote service through OAuth.

4.70.2 Constructor & Destructor Documentation

4.70.2.1 SynchronizationManager()

```
quentier::SynchronizationManager::SynchronizationManager (
            QString host,
            LocalStorageManagerAsync & localStorageManagerAsync,
            IAuthenticationManager & authenticationManager,
            QObject * parent = nullptr,
            INoteStorePtr pNoteStore = {},
            IUserStorePtr pUserStore = {},
            IKeychainServicePtr pKeychainService = {},
            ISyncStateStoragePtr pSyncStateStorage = {} )
```

Parameters

host	The host to use for the connection with the Evernote service - typically www.evernote.com but could also be sandbox.evernote.com or some other one
localStorageManagerAsync	Local storage manager
authenticationManager	Reference to an object implementing IAuthenticationManager interface; SynchronizationManager does not store this reference, it only connects to the object via signals and slots during construction

Parameters

pNoteStore	Pointer to an object implementing INoteStore interface; if nullptr, SynchronizatrionManager would create and use its own instance; otherwise SynchronizationManager would set itself as the parent of the passed in object
pUserStore	Pointer to an object implementing IUserStore interface; if nullptr, SynchronizationManager would create and use its own instance
pKeychainService	Pointer to an object implementing IKeychainService interface; if nullptr, SynchronizationManager would create and use its own default instance; otherwise SynchronizationManager would set itself as the parent of the passed in object
pSyncStateStorage	Pointer to an object implementing ISyncStateStorage interface; if nullptr, SynchronizationManager would create and use its own instance; otherwise SynchronizationManager would set itself as the parent of the passed in object

4.70.3 Member Function Documentation

4.70.3.1 active()

```
bool quentier::SynchronizationManager::active ( ) const
```

Returns

True if the synchronization is being performed at the moment, false otherwise

4.70.3.2 authenticate

```
void quentier::SynchronizationManager::authenticate ( )  [slot]
```

Use this slot to authenticate the new user to do the synchronization with the Evernote service via the client app. The invocation of this slot would respond asynchronously with authenticationFinished signal but won't start the synchronization.

Note that this slot would always proceed to the actual OAuth.

4.70.3.3 authenticateCurrentAccount

```
void quentier::SynchronizationManager::authenticateCurrentAccount ( )  [slot]
```

Use this slot to authenticate the current account to do the synchronization with the Evernote service via the client app. The invocation of this slot would respond asynchronously with authenticationFinished signal but won't start the synchronization

If no account was set to SynchronizationManager prior to this slot invocation, it would proceed to OAuth. Otherwise SynchronizationManager would first check whether the persistent authentication data is in place and actual. If yes, no OAuth would be performed.

4.70.3.4 authenticationFinished

```
void quentier::SynchronizationManager::authenticationFinished (
            bool success,
            ErrorString errorDescription,
            Account account )  [signal]
```

This signal is emitted in response to the explicit attempt to authenticate the new user of the client app to the Evernote service. NOTE: this signal is not emitted if the authentication was requested automatically during sync attempt, it is only emitted in response to the explicit invokation of authenticate slot.

Parameters

success	True if the authentication was successful, false otherwise
errorDescription	The textual explanation of the failure to authenticate the new user
account	The account of the authenticated user

4.70.3.5 authenticationRevoked

```
void quentier::SynchronizationManager::authenticationRevoked (
            bool success,
            ErrorString errorDescription,
            qevercloud::UserID userId )  [signal]
```

This signal is emitted in response to the attempt to revoke the authentication for a given user ID

Parameters

success	True if the authentication was revoked successfully, false otherwise
errorDescription	The textual explanation of the failure to revoke the authentication
userId	The ID of the user for which the revoke of the authentication was requested

4.70.3.6 detectedConflictDuringLocalChangesSending

```
void quentier::SynchronizationManager::detectedConflictDuringLocalChangesSending ( )  [signal]
```

This signal is emitted if during the "send local changes" synchronization step it was found out that new changes from the Evernote service are available AND some of them conflict with the local changes being sent.

Such situation can rarely happen in case of changes introduced concurrently with the running synchronization - perhaps via another client. The algorithm will handle it by repeating the "remote to local" incremental synchronization step, the signal is just for the sake of diagnostic.

4.70.3.7 downloadNoteThumbnailsOption()

```
bool quentier::SynchronizationManager::downloadNoteThumbnailsOption ( ) const
```

Returns

True or false depending on the option to download the thumbnails for notes containing resources during sync; by default no thumbnails are downloaded

4.70.3.8 failed

```
void quentier::SynchronizationManager::failed (
            ErrorString errorDescription )  [signal]
```

This signal is emitted when the synchronization fails; at this moment there is no error code explaining the reason of the failure programmatically so the only explanation available is the textual one for the end user

4.70.3.9 finished

```
void quentier::SynchronizationManager::finished (
            Account account,
            bool somethingDownloaded,
            bool somethingSent )  [signal]
```

This signal is emitted when the synchronization is finished

Parameters

account	Represents the latest version of Account structure filled during the synchronization procedure
somethingDownloaded	Boolean parameter telling the receiver whether any data items were actually downloaded during remote to local synchronization step; if there was nothing to sync up from the remote storage, this boolean would be false, otherwise it would be true
somethingSent	Boolean parameter telling the receiver whether any data items were actually sent during the send local changes synchronization step; if there was nothing to send to the remote storage, this boolean would be false, otherwise it would be true

4.70.3.10 linkedNotebooksNotesDownloadProgress

```
void quentier::SynchronizationManager::linkedNotebooksNotesDownloadProgress (
            quint32 notesDownloaded,
            quint32 totalNotesToDownload )  [signal]
```

This signal is emitted on each successful download of full note data from linked notebooks.

Parameters

notesDownloaded	The number of notes downloaded by the moment
totalNotesToDownload	The total number of notes that need to be downloaded

4.70.3.11 linkedNotebooksResourcesDownloadProgress

```
void quentier::SynchronizationManager::linkedNotebooksResourcesDownloadProgress (
            quint32 resourcesDownloaded,
            quint32 totalResourcesToDownload )  [signal]
```

This signal is emitted on each successful download of full resource data from linked notebooks during the incremental sync (as individual resources are downloaded along with their notes during full sync).

Parameters

resourcesDownloaded	The number of resources downloaded by the moment
totalResourcesToDownload	The total number of resources that need to be downloaded

4.70.3.12 linkedNotebooksSyncChunksDownloaded

```
void quentier::SynchronizationManager::linkedNotebooksSyncChunksDownloaded ( )  [signal]
```

This signal is emitted when the sync chunks for the stuff from linked notebooks are downloaded during "remote to local" synchronization step

4.70.3.13 linkedNotebookSyncChunksDataProcessingProgress

```
void quentier::SynchronizationManager::linkedNotebookSyncChunksDataProcessingProgress (
            ISyncChunksDataCountersPtr counters )  [signal]
```

This signal is emitted during linked notebooks' downloaded sync chunks contents processing and denotes the progress on that step.

4.70.3.14 linkedNotebookSyncChunksDownloadProgress

```
void quentier::SynchronizationManager::linkedNotebookSyncChunksDownloadProgress (
            qint32 highestDownloadedUsn,
            qint32 highestServerUsn,
            qint32 lastPreviousUsn,
            LinkedNotebook linkedNotebook )  [signal]
```

This signal is emitted during linked notebooks sync chunks downloading and denotes the progress of that step, individually for each linked notebook. The percentage of completeness can be computed roughly as (highest↩ DownloadedUsn - lastPreviousUsn) / (highestServerUsn - lastPreviousUsn) $*$ 100%. The sync chunks for each linked notebook are downloaded sequentially so the signals for one linked notebook should not intermix with signals for other linked notebooks, however, it is within hands of Qt's slot schedulers

Parameters

highestDownloadedUsn	The highest update sequence number within data items from linked notebook sync chunks downloaded so far
highestServerUsn	The current highest update sequence number within the linked notebook
lastPreviousUsn	The last update sequence number from previous sync of the given linked notebook; if current sync is the first one, this value is zero
linkedNotebook	The linked notebook which sync chunks download progress is reported

4.70.3.15 notesDownloadProgress

```
void quentier::SynchronizationManager::notesDownloadProgress (
            quint32 notesDownloaded,
            quint32 totalNotesToDownload )  [signal]
```

This signal is emitted on each successful download of full note data from use 's own account.

Parameters

notesDownloaded	The number of notes downloaded by the moment
totalNotesToDownload	The total number of notes that need to be downloaded

4.70.3.16 preparedDirtyObjectsForSending

```
void quentier::SynchronizationManager::preparedDirtyObjectsForSending ( )  [signal]
```

This signal is emitted during "send local changes" synchronization step when all the relevant data elements from user's own account were prepared for sending to the Evernote service

4.70.3.17 preparedLinkedNotebooksDirtyObjectsForSending

```
void quentier::SynchronizationManager::preparedLinkedNotebooksDirtyObjectsForSending ( )  [signal]
```

This signal is emitted during "send local changes" synchronization step when all the relevant data elements from linked notebooks were prepared for sending to the Evernote service

4.70.3.18 rateLimitExceeded

```
void quentier::SynchronizationManager::rateLimitExceeded (
            qint32 secondsToWait )  [signal]
```

This signal is emitted when the Evernote API rate limit is breached during the synchronization; the algorithm will handle it by auto-pausing itself until the time necessary to wait passes and then automatically continuing the synchronization.

Parameters

secondsToWait	The amount of time (in seconds) necessary to wait before the synchronization will continue

4.70.3.19 remoteToLocalSyncDone

```
void quentier::SynchronizationManager::remoteToLocalSyncDone (
            bool somethingDownloaded )  [signal]
```

This signal is emitted when the "remote to local" synchronization step is finished; once that step is done, the algorithn switches to sending the local changes back to the Evernote service.

Parameters

somethingDownloaded	Boolean parameter telling the receiver whether any data items were actually downloaded during remote to local synchronization step; if there was nothing to sync up from the remote storage, this boolean would be false, otherwise it would be true

4.70.3.20 remoteToLocalSyncStopped

```
void quentier::SynchronizationManager::remoteToLocalSyncStopped ( )  [signal]
```

This signal is emitted when the "remote to local" synchronization step is stopped

4.70.3.21 resourcesDownloadProgress

```
void quentier::SynchronizationManager::resourcesDownloadProgress (
            quint32 resourcesDownloaded,
            quint32 totalResourcesToDownload )  [signal]
```

This signal is emitted on each successful doenload of full resource data from user's own account during the incremental sync (as individual resources are downloaded along with their notes during full sync).

Parameters

resourcesDownloaded	The number of resources downloaded by the moment
totalResourcesToDownload	The total number of resources that need to be downloaded

4.70.3.22 revokeAuthentication

```
void quentier::SynchronizationManager::revokeAuthentication (
            const qevercloud::UserID userId )  [slot]
```

Use this slot to remove any previously cached authentication tokens (and shard ids) for a given user ID. After the call of this method the next attempt to synchronize the data for this user ID would cause the launch of OAuth to get the new authentication token

4.70.3.23 sendLocalChangesStopped

```
void quentier::SynchronizationManager::sendLocalChangesStopped ( )  [signal]
```

This signal is emitted when the "send local changes" synchronization step is stopped

4.70.3.24 setAccount

```
void quentier::SynchronizationManager::setAccount (
            Account account )  [slot]
```

Use this slot to set the current account for the synchronization manager. If the slot is called during the synchronization running, it would stop, any internal caches belonging to previously selected account (if any) would be purged (but persistent settings like the authentication token saved in the system keychain would remain). Setting the current account won't automatically start the synchronization for it, use synchronize slot for this.

The attempt to set the current account of "Local" type would just clean up the synchronization manager as if it was just created

After the method finishes its job, setAccountDone signal is emitted

4.70.3.25 setAccountDone

```
void quentier::SynchronizationManager::setAccountDone (
            Account account )  [signal]
```

This signal is emitted in response to invoking the setAccount slot after all the activities involved in switching the account inside SynchronizationManager are finished

4.70.3.26 setDownloadInkNoteImages

```
void quentier::SynchronizationManager::setDownloadInkNoteImages (
            bool flag )  [slot]
```

Use this slot to switch the option whether the synchronization of notes would download the plain images corresponding to ink notes or not. By default the downloading of ink note images is disabled.

After the method finishes its job, setDownloadInkNoteImagesDone signal is emitted

4.70.3.27 setDownloadInkNoteImagesDone

```
void quentier::SynchronizationManager::setDownloadInkNoteImagesDone (
            bool flag )  [signal]
```

This signal is emitted in response to invoking the setDownloadInkNoteImages slot after the setting is accepted

4.70.3.28 setDownloadNoteThumbnails

```
void quentier::SynchronizationManager::setDownloadNoteThumbnails (
            bool flag )  [slot]
```

Use this slot to switch the option whether the synchronization of notes would download the note thumbnails or not. By default the downloading of thumbnails for notes containing resources is disabled.

NOTE: even if thumbnails downloading is enabled, the thumbnails would be downloaded during sync only for notes containing resources

After the method finishes its job, setDownloadNoteThumbnailsDone signal is emitted

4.70.3.29 setDownloadNoteThumbnailsDone

```
void quentier::SynchronizationManager::setDownloadNoteThumbnailsDone (
            bool flag )  [signal]
```

This signal is emitted in response to invoking the setDownloadNoteThumbnails slot after the setting is accepted

4.70.3.30 setInkNoteImagesStoragePath

```
void quentier::SynchronizationManager::setInkNoteImagesStoragePath (
            QString path )  [slot]
```

Use this slot to specify the path to folder at which the downloaded ink note images should be stored. Each ink note image would be stored in a separate PNG file which name would be the same as the guid of the corresponding resource and the file extension would be PNG

The default storage path would be the folder "inkNoteImages" within the folder returned by applicationPersistent↩ StoragePath function found in quentier/StandardPaths.h header

WARNING: if the passed in path cannot be used (either it doesn't exist and cannot be created or exists but is not writable), the default path is silently restored. So make sure you're setting a valid path

After the method finishes its job, setInkNoteImagesStoragePathDone signal is emitted

4.70.3.31 setInkNoteImagesStoragePathDone

```
void quentier::SynchronizationManager::setInkNoteImagesStoragePathDone (
            QString path )  [signal]
```

This signal is emitted in response to invoking the setInkNoteImagesStoragePath slot after the setting is accepted

4.70.3.32 started

```
void quentier::SynchronizationManager::started ( )  [signal]
```

This signal is emitted when the synchronization is started (authentication is not considered a part of synchronization so this signal is only emitted when the authentication is completed)

4.70.3.33 stop

```
void quentier::SynchronizationManager::stop ( )  [slot]
```

Use this slot to stop the running synchronization; if no synchronization is running by the moment of this slot call, it has no effect

4.70.3.34 stopped

```
void quentier::SynchronizationManager::stopped ( )  [signal]
```

This signal is emitted in response to invoking the stop slot, whether it was invoked manually or from within the SynchronizationManager itself (due to sync failure, for example)

4.70.3.35 syncChunksDataProcessingProgress

```
void quentier::SynchronizationManager::syncChunksDataProcessingProgress (
            ISyncChunksDataCountersPtr counters )  [signal]
```

This signal is emitted during user own account's downloaded sync chunks contents processing and denotes the progress on that step.

4.70.3.36 syncChunksDownloaded

```
void quentier::SynchronizationManager::syncChunksDownloaded ( )  [signal]
```

This signal is emitted when the sync chunks for the stuff from user's own account are downloaded during "remote to local" synchronization step

4.70.3.37 syncChunksDownloadProgress

```
void quentier::SynchronizationManager::syncChunksDownloadProgress (
            qint32 highestDownloadedUsn,
            qint32 highestServerUsn,
            qint32 lastPreviousUsn )  [signal]
```

This signal is emitted during user own account's sync chunks downloading and denotes the progress of that step. The percentage of completeness can be computed roughly as (highestDownloadedUsn - lastPreviousUsn) / (highestServerUsn - lastPreviousUsn) $*$ 100%

Parameters

highestDownloadedUsn	The highest update sequence number within data items from sync chunks downloaded so far
highestServerUsn	The current highest update sequence number within the account
lastPreviousUsn	The last update sequence number from previous sync; if current sync is the first one, this value is zero

4.70.3.38 synchronize

```
void quentier::SynchronizationManager::synchronize ( )  [slot]
```

Use this slot to launch the synchronization of data

4.70.3.39 willRepeatRemoteToLocalSyncAfterSendingChanges

```
void quentier::SynchronizationManager::willRepeatRemoteToLocalSyncAfterSendingChanges ( )
[signal]
```

This signal is emitted if during the "send local changes" synchronization step it was found out that new changes from the Evernote service are available yet no conflict between remote and local changes was found yet.

Such situation can rarely happen in case of changes introduced concurrently with the running synchronization - perhaps via another client. The algorithm will handle it, the signal is just for the sake of diagnostic.

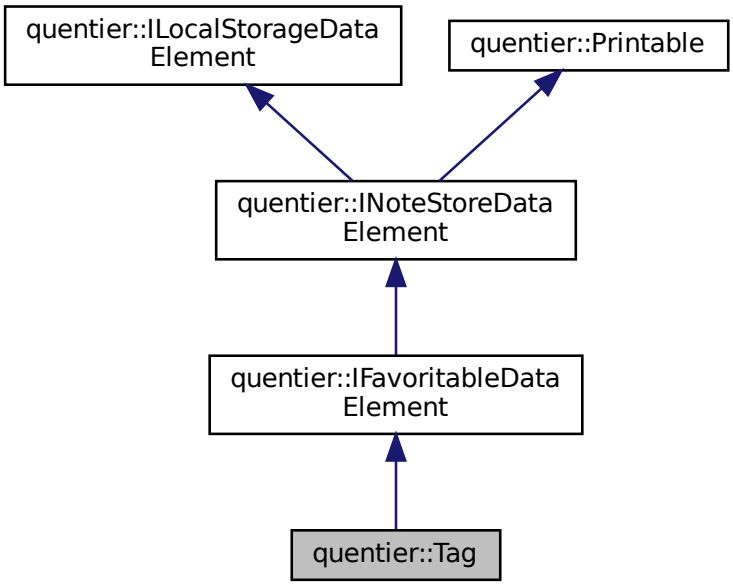
4.71 quentier::SysInfo Class Reference

Public Member Functions

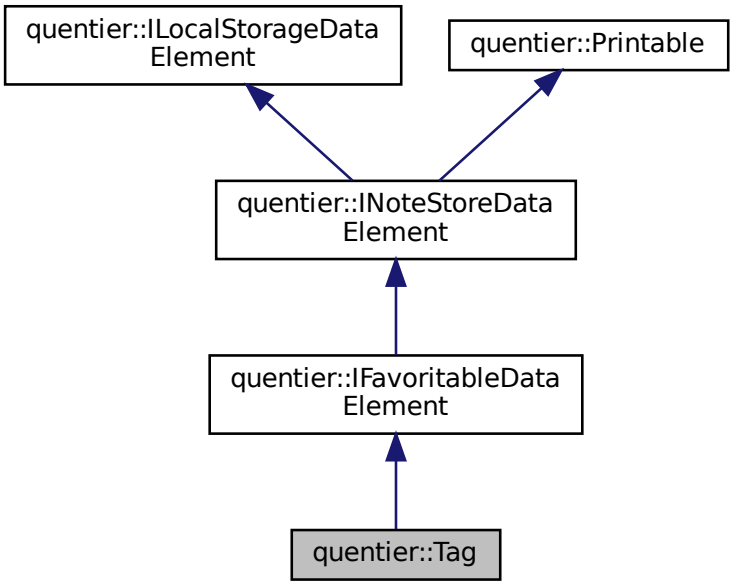
- qint64 **pageSize** ()
- qint64 **totalMemory** ()
- qint64 **freeMemory** ()
- QString **stackTrace** ()
- QString **platformName** ()

4.72 quentier::Tag Class Reference

Inheritance diagram for quentier::Tag:



Collaboration diagram for quentier::Tag:



Public Member Functions

- **Tag** (const Tag &other)
- **Tag** (Tag &&other)
- Tag & **operator=** (const Tag &other)
- Tag & **operator=** (Tag &&other)
- **Tag** (const qevercloud::Tag &other)
- **Tag** (qevercloud::Tag &&other)
- bool **operator==** (const Tag &other) const
- bool **operator!=** (const Tag &other) const
- const qevercloud::Tag & **qevercloudTag** () const
- qevercloud::Tag & **qevercloudTag** ()
- virtual void **clear** () override
- virtual bool **hasGuid** () const override
- virtual const QString & **guid** () const override
- virtual void **setGuid** (const QString &guid) override
- virtual bool **hasUpdateSequenceNumber** () const override
- virtual qint32 **updateSequenceNumber** () const override
- virtual void **setUpdateSequenceNumber** (const qint32 usn) override
- virtual bool **checkParameters** (ErrorString &errorDescription) const override
- bool **hasName** () const
- const QString & **name** () const
- void **setName** (const QString &name)
- bool **hasParentGuid** () const
- const QString & **parentGuid** () const
- void **setParentGuid** (const QString &parentGuid)
- bool **hasParentLocalUid** () const

- const QString & **parentLocalUid** () const
- void **setParentLocalUid** (const QString &parentLocalUid)
- bool **hasLinkedNotebookGuid** () const
- const QString & **linkedNotebookGuid** () const
- void **setLinkedNotebookGuid** (const QString &linkedNotebookGuid)
- virtual QTextStream & **print** (QTextStream &strm) const override

Static Public Member Functions

- static bool **validateName** (const QString &name, ErrorString *pErrorDescription=nullptr)

Additional Inherited Members

4.73 quentier::ResourceRecognitionIndexItem::TextItem Struct Reference

Public Member Functions

- bool **operator==** (const TextItem &other) const

Public Attributes

- QString **m_text**
- int **m_weight** = -1

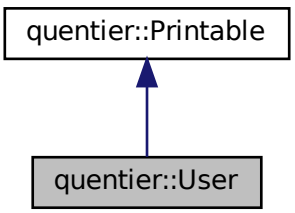
4.74 quentier::UidGenerator Class Reference

Static Public Member Functions

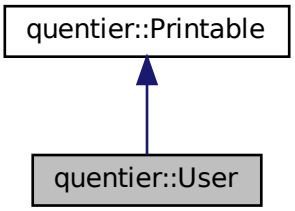
- static QString **Generate** ()
- static QString **UidToString** (const QUuid &uid)

4.75 quentier::User Class Reference

Inheritance diagram for quentier::User:



Collaboration diagram for quentier::User:



Public Types

- using **PrivilegeLevel** = qevercloud::PrivilegeLevel
- using **ServiceLevel** = qevercloud::ServiceLevel

Public Member Functions

- **User** (const qevercloud::User &user)
- **User** (qevercloud::User &&user)
- **User** (const User &other)
- **User** (User &&other)
- User & **operator=** (const User &other)
- User & **operator=** (User &&other)
- bool **operator==** (const User &other) const
- bool **operator!=** (const User &other) const
- const qevercloud::User & **qevercloudUser** () const
- qevercloud::User & **qevercloudUser** ()
- void **clear** ()
- bool **isDirty** () const
- void **setDirty** (const bool dirty)
- bool **isLocal** () const
- void **setLocal** (const bool local)
- bool **checkParameters** (ErrorString &errorDescription) const
- bool **hasId** () const
- qint32 **id** () const
- void **setId** (const qint32 id)
- bool **hasUsername** () const
- const QString & **username** () const
- void **setUsername** (const QString &username)
- bool **hasEmail** () const
- const QString & **email** () const
- void **setEmail** (const QString &email)
- bool **hasName** () const
- const QString & **name** () const
- void **setName** (const QString &name)
- bool **hasTimezone** () const

- const QString & **timezone** () const
- void **setTimezone** (const QString &timezone)
- bool **hasPrivilegeLevel** () const
- PrivilegeLevel **privilegeLevel** () const
- void **setPrivilegeLevel** (const qint8 level)
- bool **hasServiceLevel** () const
- ServiceLevel **serviceLevel** () const
- void **setServiceLevel** (const qint8 level)
- bool **hasCreationTimestamp** () const
- qint64 **creationTimestamp** () const
- void **setCreationTimestamp** (const qint64 timestamp)
- bool **hasModificationTimestamp** () const
- qint64 **modificationTimestamp** () const
- void **setModificationTimestamp** (const qint64 timestamp)
- bool **hasDeletionTimestamp** () const
- qint64 **deletionTimestamp** () const
- void **setDeletionTimestamp** (const qint64 timestamp)
- bool **hasActive** () const
- bool **active** () const
- void **setActive** (const bool active)
- bool **hasShardId** () const
- const QString & **shardId** () const
- void **setShardId** (const QString &shardId)
- bool **hasUserAttributes** () const
- const qevercloud::UserAttributes & **userAttributes** () const
- void **setUserAttributes** (qevercloud::UserAttributes &&attributes)
- bool **hasAccounting** () const
- const qevercloud::Accounting & **accounting** () const
- void **setAccounting** (qevercloud::Accounting &&accounting)
- bool **hasBusinessUserInfo** () const
- const qevercloud::BusinessUserInfo & **businessUserInfo** () const
- void **setBusinessUserInfo** (qevercloud::BusinessUserInfo &&info)
- bool **hasPhotoUrl** () const
- QString **photoUrl** () const
- void **setPhotoUrl** (const QString &photoUrl)
- bool **hasPhotoLastUpdateTimestamp** () const
- qint64 **photoLastUpdateTimestamp** () const
- void **setPhotoLastUpdateTimestamp** (const qint64 timestamp)
- bool **hasAccountLimits** () const
- const qevercloud::AccountLimits & **accountLimits** () const
- void **setAccountLimits** (qevercloud::AccountLimits &&limits)
- virtual QTextStream & **print** (QTextStream &strm) const override

Friends

- class **Notebook**

Additional Inherited Members

Index

~ApplicationSettings
quentier::ApplicationSettings, 15

accountHighUsn
quentier::LocalStorageManager, 107
active
quentier::SynchronizationManager, 199
addEnResource
quentier::LocalStorageManager, 108
addLinkedNotebook
quentier::LocalStorageManager, 108
addNote
quentier::LocalStorageManager, 109
addNotebook
quentier::LocalStorageManager, 109
addSavedSearch
quentier::LocalStorageManager, 110
addTag
quentier::LocalStorageManager, 110
addUser
quentier::LocalStorageManager, 110
addedLinkedNotebooks
quentier::ISyncChunksDataCounters, 85
addedNotebooks
quentier::ISyncChunksDataCounters, 85
addedSavedSearches
quentier::ISyncChunksDataCounters, 85
addedTags
quentier::ISyncChunksDataCounters, 86
ApplicationSettings
quentier::ApplicationSettings, 14, 15
apply
quentier::ILocalStoragePatch, 62
authenticate
quentier::SynchronizationManager, 199
authenticateCurrentAccount
quentier::SynchronizationManager, 199
authenticateToSharedNotebook
quentier::INoteStore, 71
authenticationFinished
quentier::SynchronizationManager, 199
authenticationRevoked
quentier::SynchronizationManager, 200

backend
quentier::NoteEditor, 161
backupLocalStorage
quentier::ILocalStoragePatch, 62
backupProgress
quentier::ILocalStoragePatch, 63

beginGroup
quentier::ApplicationSettings, 15, 16
beginReadArray
quentier::ApplicationSettings, 16
beginWriteArray
quentier::ApplicationSettings, 17

checkLinkedNotebooks
quentier::DefaultLocalStorageCacheExpiry↩
Checker, 31
quentier::ILocalStorageCacheExpiryChecker, 59
checkNotebooks
quentier::DefaultLocalStorageCacheExpiry↩
Checker, 32
quentier::ILocalStorageCacheExpiryChecker, 59
checkNotes
quentier::DefaultLocalStorageCacheExpiry↩
Checker, 32
quentier::ILocalStorageCacheExpiryChecker, 59
checkResources
quentier::DefaultLocalStorageCacheExpiry↩
Checker, 32
quentier::ILocalStorageCacheExpiryChecker, 59
checkSavedSearches
quentier::DefaultLocalStorageCacheExpiry↩
Checker, 32
quentier::ILocalStorageCacheExpiryChecker, 59
checkTags
quentier::DefaultLocalStorageCacheExpiry↩
Checker, 33
quentier::ILocalStorageCacheExpiryChecker, 60
checkVersion
quentier::IUserStore, 91
cleanupExternalHtml
quentier::ENMLConverter, 37
clear
quentier::NoteEditor, 162
clone
quentier::DefaultLocalStorageCacheExpiry↩
Checker, 33
quentier::ILocalStorageCacheExpiryChecker, 60
contains
quentier::ApplicationSettings, 17, 18
convertToNote
quentier::NoteEditor, 162
createNote
quentier::INoteStore, 71
createNotebook
quentier::INoteStore, 72
createSavedSearch

quentier::INoteStore, 72
createTag
quentier::INoteStore, 73
currentNoteLocalUid
quentier::NoteEditor, 162

defaultFont
quentier::NoteEditor, 162
defaultPalette
quentier::NoteEditor, 162
defaultShortcut
quentier::ShortcutManager, 191, 192
deletePasswordJobFinished
quentier::IKeychainService, 53
deleteUser
quentier::LocalStorageManager, 111
detectedConflictDuringLocalChangesSending
quentier::SynchronizationManager, 200
displayName
quentier::Account, 11
downloadNoteThumbnailsOption
quentier::SynchronizationManager, 200

enResourceCount
quentier::LocalStorageManager, 111
ErrorCode
quentier::IKeychainService, 53
evernoteAccountType
quentier::Account, 11
evernoteHost
quentier::Account, 11
exportNotesToEnex
quentier::ENMLConverter, 38
expungeEnResource
quentier::LocalStorageManager, 112
expungeLinkedNotebook
quentier::LocalStorageManager, 112
expungeNote
quentier::LocalStorageManager, 113
expungeNotebook
quentier::LocalStorageManager, 113
expungeNotelessTagsFromLinkedNotebooks
quentier::LocalStorageManager, 114
expungeSavedSearch
quentier::LocalStorageManager, 114
expungeTag
quentier::LocalStorageManager, 114
expungeUser
quentier::LocalStorageManager, 115
expungedLinkedNotebooks
quentier::ISyncChunksDataCounters, 86
expungedNotebooks
quentier::ISyncChunksDataCounters, 86
expungedSavedSearches
quentier::ISyncChunksDataCounters, 86
expungedTags
quentier::ISyncChunksDataCounters, 86

failed
quentier::SynchronizationManager, 201
findDefaultNotebook
quentier::LocalStorageManager, 115
findDefaultOrLastUsedNotebook
quentier::LocalStorageManager, 116
findEnResource
quentier::LocalStorageManager, 116
findLastUsedNotebook
quentier::LocalStorageManager, 117
findLinkedNotebook
quentier::LocalStorageManager, 117
findNote
quentier::LocalStorageManager, 117
findNoteLocalUidsWithSearchQuery
quentier::LocalStorageManager, 118
findNotebook
quentier::LocalStorageManager, 118
findNotesWithSearchQuery
quentier::LocalStorageManager, 119
findSavedSearch
quentier::LocalStorageManager, 119
findTag
quentier::LocalStorageManager, 120
findUser
quentier::LocalStorageManager, 120
finished
quentier::SynchronizationManager, 201
fromVersion
quentier::ILocalStoragePatch, 63

getAccountLimits
quentier::IUserStore, 92
getLinkedNotebookSyncChunk
quentier::INoteStore, 73
getLinkedNotebookSyncState
quentier::INoteStore, 74
getNote
quentier::INoteStore, 75
getNoteAsync
quentier::INoteStore, 75
GetNoteOption
quentier::LocalStorageManager, 105
getResource
quentier::INoteStore, 76
getResourceAsync
quentier::INoteStore, 77
GetResourceOption
quentier::LocalStorageManager, 106
getSyncChunk
quentier::INoteStore, 77
getSyncState
quentier::INoteStore, 78
getUser
quentier::IUserStore, 92

highestSupportedLocalStorageVersion
quentier::LocalStorageManager, 121
htmlToQTextDocument
quentier::ENMLConverter, 38

id
quentier::Account, 11
idleTime
quentier::NoteEditor, 162
importEnex
quentier::ENMLConverter, 39
inAppNoteLinkPasteRequested
quentier::NoteEditor, 163
initialize
quentier::NoteEditor, 163
isEditorPageModified
quentier::NoteEditor, 163
isEmpty
quentier::Account, 11
isLocalStorageVersionTooHigh
quentier::LocalStorageManager, 121
isModified
quentier::NoteEditor, 164
isNoteLoaded
quentier::NoteEditor, 164

linkedNotebookCount
quentier::LocalStorageManager, 121
linkedNotebookSyncChunksDataProcessingProgress
quentier::SynchronizationManager, 202
linkedNotebookSyncChunksDownloadProgress
quentier::SynchronizationManager, 202
linkedNotebooksNotesDownloadProgress
quentier::SynchronizationManager, 201
linkedNotebooksResourcesDownloadProgress
quentier::SynchronizationManager, 202
linkedNotebooksSyncChunksDownloaded
quentier::SynchronizationManager, 202
listAllLinkedNotebooks
quentier::LocalStorageManager, 122
listAllNotebooks
quentier::LocalStorageManager, 122
listAllSavedSearches
quentier::LocalStorageManager, 123
listAllSharedNotebooks
quentier::LocalStorageManager, 124
listAllTags
quentier::LocalStorageManager, 124
listAllTagsPerNote
quentier::LocalStorageManager, 125
listLinkedNotebooks
quentier::LocalStorageManager, 125
listNotebooks
quentier::LocalStorageManager, 126
listNotes
quentier::LocalStorageManager, 127
listNotesByLocalUids
quentier::LocalStorageManager, 127
listNotesPerNotebook
quentier::LocalStorageManager, 128
listNotesPerNotebooksAndTags
quentier::LocalStorageManager, 129
listNotesPerTag
quentier::LocalStorageManager, 129
ListObjectsOption
quentier::LocalStorageManager, 106
listSavedSearches
quentier::LocalStorageManager, 130
listSharedNotebooksPerNotebookGuid
quentier::LocalStorageManager, 131
listTags
quentier::LocalStorageManager, 131
listTagsWithNoteLocalUids
quentier::LocalStorageManager, 132
LocalStorageManager
quentier::LocalStorageManager, 107
localStorageRequiresUpgrade
quentier::LocalStorageManager, 133
localStorageVersion
quentier::LocalStorageManager, 133

name
quentier::Account, 12
noteCount
quentier::LocalStorageManager, 134
noteCountPerNotebook
quentier::LocalStorageManager, 134
noteCountPerNotebooksAndTags
quentier::LocalStorageManager, 135
noteCountPerTag
quentier::LocalStorageManager, 135
noteCountsPerAllTags
quentier::LocalStorageManager, 136
noteStoreUrl
quentier::INoteStore, 79
notebookCount
quentier::LocalStorageManager, 134
notebookModifier
quentier::NoteSearchQuery, 172
notesDownloadProgress
quentier::SynchronizationManager, 203
notifySyncStateUpdated
quentier::ISyncStateStorage, 90

onReadFileRequest
quentier::FileIOProcessorAsync, 45
onWriteFileRequest
quentier::FileIOProcessorAsync, 45
Option
quentier::DateTimePrint, 29

patchLongDescription
quentier::ILocalStoragePatch, 63
patchShortDescription
quentier::ILocalStoragePatch, 63
preparedDirtyObjectsForSending
quentier::SynchronizationManager, 203
preparedLinkedNotebooksDirtyObjectsForSending
quentier::SynchronizationManager, 203
progress
quentier::ILocalStoragePatch, 64

quentier::Account, 9

displayName, 11
evernoteAccountType, 11
evernoteHost, 11
id, 11
isEmpty, 11
name, 12
setDisplayName, 12
shardId, 12
type, 12
quentier::ApplicationSettings, 13
~ApplicationSettings, 15
ApplicationSettings, 14, 15
beginGroup, 15, 16
beginReadArray, 16
beginWriteArray, 17
contains, 17, 18
remove, 18
setValue, 19
value, 20
quentier::ApplicationSettings::ArrayCloser, 22
quentier::ApplicationSettings::GroupCloser, 48
quentier::ApplicationSettingsInitializationException, 20
quentier::AuthenticationManager, 23
quentier::DatabaseLockFailedException, 25
quentier::DatabaseLockedException, 24
quentier::DatabaseOpeningException, 26
quentier::DatabaseRequestException, 27
quentier::DateTimePrint, 29
Option, 29
quentier::DecryptedTextManager, 29
quentier::DefaultLocalStorageCacheExpiryChecker, 30
checkLinkedNotebooks, 31
checkNotebooks, 32
checkNotes, 32
checkResources, 32
checkSavedSearches, 32
checkTags, 33
clone, 33
quentier::ENMLConverter, 36
cleanupExternalHtml, 37
exportNotesToEnex, 38
htmlToQTextDocument, 38
importEnex, 39
quentier::ENMLConverter::NoteContentToHtmlExtra↩
Data, 158
quentier::ENMLConverter::SkipHtmlElementRule, 193
quentier::EmptyDataElementException, 34
quentier::EncryptionManager, 35
quentier::ErrorString, 39
quentier::EventLoopWithExitStatus, 41
quentier::FileCopier, 42
quentier::FileIOProcessorAsync, 44
onReadFileRequest, 45
onWriteFileRequest, 45
readFileRequestProcessed, 46
setIdleTimePeriod, 46
writeFileRequestProcessed, 46
quentier::FileSystemWatcher, 47
quentier::HTMLCleaner, 49
quentier::IAuthenticationManager, 49
quentier::IFavoritableDataElement, 50
quentier::IKeychainService, 51
deletePasswordJobFinished, 53
ErrorCode, 53
readPasswordJobFinished, 55
startDeletePasswordJob, 55
startReadPasswordJob, 55
startWritePasswordJob, 56
writePasswordJobFinished, 56
quentier::ILocalStorageCacheExpiryChecker, 57
checkLinkedNotebooks, 59
checkNotebooks, 59
checkNotes, 59
checkResources, 59
checkSavedSearches, 59
checkTags, 60
clone, 60
quentier::ILocalStorageDataElement, 60
quentier::ILocalStoragePatch, 61
apply, 62
backupLocalStorage, 62
backupProgress, 63
fromVersion, 63
patchLongDescription, 63
patchShortDescription, 63
progress, 64
removeLocalStorageBackup, 64
restoreBackupProgress, 64
restoreLocalStorageFromBackup, 65
toVersion, 65
quentier::INoteEditorBackend, 66
quentier::INoteStore, 69
authenticateToSharedNotebook, 71
createNote, 71
createNotebook, 72
createSavedSearch, 72
createTag, 73
getLinkedNotebookSyncChunk, 73
getLinkedNotebookSyncState, 74
getNote, 75
getNoteAsync, 75
getResource, 76
getResourceAsync, 77
getSyncChunk, 77
getSyncState, 78
noteStoreUrl, 79
setAuthData, 79
setNoteStoreUrl, 79
stop, 79
updateNote, 79
updateNotebook, 80
updateSavedSearch, 80
updateTag, 81
quentier::INoteStoreDataElement, 81
quentier::IQuentierException, 82
quentier::ISyncChunksDataCounters, 84

addedLinkedNotebooks, 85
addedNotebooks, 85
addedSavedSearches, 85
addedTags, 86
expungedLinkedNotebooks, 86
expungedNotebooks, 86
expungedSavedSearches, 86
expungedTags, 86
totalExpungedLinkedNotebooks, 86
totalExpungedNotebooks, 86
totalExpungedSavedSearches, 87
totalExpungedTags, 87
totalLinkedNotebooks, 87
totalNotebooks, 87
totalSavedSearches, 87
totalTags, 87
updatedLinkedNotebooks, 87
updatedNotebooks, 88
updatedSavedSearches, 88
updatedTags, 88
quentier::ISyncStateStorage, 89
notifySyncStateUpdated, 90
quentier::ISyncStateStorage::ISyncState, 88
quentier::IUserStore, 91
checkVersion, 91
getAccountLimits, 92
getUser, 92
setAuthData, 93
quentier::LRUCache< Key, Value, Allocator >, 150
quentier::LimitedStack< T >, 93
quentier::LinkedNotebook, 94
quentier::LocalStorageCacheManager, 96
quentier::LocalStorageCacheManagerException, 98
quentier::LocalStorageManager, 99
accountHighUsn, 107
addEnResource, 108
addLinkedNotebook, 108
addNote, 109
addNotebook, 109
addSavedSearch, 110
addTag, 110
addUser, 110
deleteUser, 111
enResourceCount, 111
expungeEnResource, 112
expungeLinkedNotebook, 112
expungeNote, 113
expungeNotebook, 113
expungeNotelessTagsFromLinkedNotebooks, 114
expungeSavedSearch, 114
expungeTag, 114
expungeUser, 115
findDefaultNotebook, 115
findDefaultOrLastUsedNotebook, 116
findEnResource, 116
findLastUsedNotebook, 117
findLinkedNotebook, 117
findNote, 117
findNoteLocalUidsWithSearchQuery, 118
findNotebook, 118
findNotesWithSearchQuery, 119
findSavedSearch, 119
findTag, 120
findUser, 120
GetNoteOption, 105
GetResourceOption, 106
highestSupportedLocalStorageVersion, 121
isLocalStorageVersionTooHigh, 121
linkedNotebookCount, 121
listAllLinkedNotebooks, 122
listAllNotebooks, 122
listAllSavedSearches, 123
listAllSharedNotebooks, 124
listAllTags, 124
listAllTagsPerNote, 125
listLinkedNotebooks, 125
listNotebooks, 126
listNotes, 127
listNotesByLocalUids, 127
listNotesPerNotebook, 128
listNotesPerNotebooksAndTags, 129
listNotesPerTag, 129
ListObjectsOption, 106
listSavedSearches, 130
listSharedNotebooksPerNotebookGuid, 131
listTags, 131
listTagsWithNoteLocalUids, 132
LocalStorageManager, 107
localStorageRequiresUpgrade, 133
localStorageVersion, 133
noteCount, 134
noteCountPerNotebook, 134
noteCountPerNotebooksAndTags, 135
noteCountPerTag, 135
noteCountsPerAllTags, 136
notebookCount, 134
requiredLocalStoragePatches, 136
savedSearchCount, 136
StartupOption, 106
switchUser, 137
tagCount, 137
updateEnResource, 138
updateLinkedNotebook, 138
updateNote, 138
UpdateNoteOption, 106
updateNotebook, 139
updateSavedSearch, 140
updateTag, 140
updateUser, 141
upgradeProgress, 141
userCount, 142
quentier::LocalStorageManagerAsync, 142
quentier::LoggerInitializationException, 149
quentier::Note, 151
quentier::NoteEditor, 158
backend, 161

clear, 162
convertToNote, 162
currentNoteLocalUid, 162
defaultFont, 162
defaultPalette, 162
idleTime, 162
inAppNoteLinkPasteRequested, 163
initialize, 163
isEditorPageModified, 163
isModified, 164
isNoteLoaded, 164
saveNoteToLocalStorage, 164
setAccount, 164
setBackend, 164
setCurrentNoteLocalUid, 164
setDefaultFont, 165
setDefaultPalette, 165
setFocus, 165
setInitialPageHtml, 165
setNoteDeletedPageHtml, 166
setNoteLoadingPageHtml, 166
setNoteNotFoundPageHtml, 166
setNoteTitle, 166
setTagIds, 166
setUndoStack, 167
undoStack, 167
quentier::NoteEditorInitializationException, 167
quentier::NoteEditorPluginInitializationException, 168
quentier::NoteSearchQuery, 169
notebookModifier, 172
queryString, 172
quentier::Notebook, 154
quentier::NullPtrException, 172
quentier::Printable, 174
quentier::QuentierApplication, 175
quentier::QuentierUndoCommand, 176
quentier::Resource, 177
quentier::ResourceRecognitionIndexItem, 180
quentier::ResourceRecognitionIndexItem::BarcodeItem, 24
quentier::ResourceRecognitionIndexItem::ObjectItem, 173
quentier::ResourceRecognitionIndexItem::ShapeItem, 185
quentier::ResourceRecognitionIndexItem::TextItem, 210
quentier::ResourceRecognitionIndices, 182
quentier::SavedSearch, 183
quentier::SharedNote, 186
quentier::SharedNotebook, 188
quentier::ShortcutManager, 190
defaultShortcut, 191, 192
shortcut, 192
userShortcut, 192, 193
quentier::SpellChecker, 195
quentier::StringUtils, 196
quentier::StringUtils::StringFilterPredicate, 196
quentier::SynchronizationManager, 196
active, 199
authenticate, 199
authenticateCurrentAccount, 199
authenticationFinished, 199
authenticationRevoked, 200
detectedConflictDuringLocalChangesSending, 200
downloadNoteThumbnailsOption, 200
failed, 201
finished, 201
linkedNotebookSyncChunksDataProcessingProgress, 202
linkedNotebookSyncChunksDownloadProgress, 202
linkedNotebooksNotesDownloadProgress, 201
linkedNotebooksResourcesDownloadProgress, 202
linkedNotebooksSyncChunksDownloaded, 202
notesDownloadProgress, 203
preparedDirtyObjectsForSending, 203
preparedLinkedNotebooksDirtyObjectsForSending, 203
rateLimitExceeded, 203
remoteToLocalSyncDone, 204
remoteToLocalSyncStopped, 204
resourcesDownloadProgress, 204
revokeAuthentication, 204
sendLocalChangesStopped, 205
setAccount, 205
setAccountDone, 205
setDownloadInkNoteImages, 205
setDownloadInkNoteImagesDone, 205
setDownloadNoteThumbnails, 205
setDownloadNoteThumbnailsDone, 206
setInkNoteImagesStoragePath, 206
setInkNoteImagesStoragePathDone, 206
started, 206
stop, 206
stopped, 207
syncChunksDataProcessingProgress, 207
syncChunksDownloadProgress, 207
syncChunksDownloaded, 207
SynchronizationManager, 198
synchronize, 207
willRepeatRemoteToLocalSyncAfterSendingChanges, 208
quentier::SysInfo, 208
quentier::Tag, 208
quentier::UidGenerator, 210
quentier::User, 210
queryString
quentier::NoteSearchQuery, 172

rateLimitExceeded
quentier::SynchronizationManager, 203
readFileRequestProcessed
quentier::FileIOProcessorAsync, 46
readPasswordJobFinished
quentier::IKeychainService, 55
remoteToLocalSyncDone
quentier::SynchronizationManager, 204

remoteToLocalSyncStopped
quentier::SynchronizationManager, 204
remove
quentier::ApplicationSettings, 18
removeLocalStorageBackup
quentier::ILocalStoragePatch, 64
requiredLocalStoragePatches
quentier::LocalStorageManager, 136
resourcesDownloadProgress
quentier::SynchronizationManager, 204
restoreBackupProgress
quentier::ILocalStoragePatch, 64
restoreLocalStorageFromBackup
quentier::ILocalStoragePatch, 65
revokeAuthentication
quentier::SynchronizationManager, 204

saveNoteToLocalStorage
quentier::NoteEditor, 164
savedSearchCount
quentier::LocalStorageManager, 136
sendLocalChangesStopped
quentier::SynchronizationManager, 205
setAccount
quentier::NoteEditor, 164
quentier::SynchronizationManager, 205
setAccountDone
quentier::SynchronizationManager, 205
setAuthData
quentier::INoteStore, 79
quentier::IUserStore, 93
setBackend
quentier::NoteEditor, 164
setCurrentNoteLocalUid
quentier::NoteEditor, 164
setDefaultFont
quentier::NoteEditor, 165
setDefaultPalette
quentier::NoteEditor, 165
setDisplayName
quentier::Account, 12
setDownloadInkNoteImages
quentier::SynchronizationManager, 205
setDownloadInkNoteImagesDone
quentier::SynchronizationManager, 205
setDownloadNoteThumbnails
quentier::SynchronizationManager, 205
setDownloadNoteThumbnailsDone
quentier::SynchronizationManager, 206
setFocus
quentier::NoteEditor, 165
setIdleTimePeriod
quentier::FileIOProcessorAsync, 46
setInitialPageHtml
quentier::NoteEditor, 165
setInkNoteImagesStoragePath
quentier::SynchronizationManager, 206
setInkNoteImagesStoragePathDone
quentier::SynchronizationManager, 206
setNoteDeletedPageHtml
quentier::NoteEditor, 166
setNoteLoadingPageHtml
quentier::NoteEditor, 166
setNoteNotFoundPageHtml
quentier::NoteEditor, 166
setNoteStoreUrl
quentier::INoteStore, 79
setNoteTitle
quentier::NoteEditor, 166
setTagIds
quentier::NoteEditor, 166
setUndoStack
quentier::NoteEditor, 167
setValue
quentier::ApplicationSettings, 19
shardId
quentier::Account, 12
shortcut
quentier::ShortcutManager, 192
startDeletePasswordJob
quentier::IKeychainService, 55
startReadPasswordJob
quentier::IKeychainService, 55
startWritePasswordJob
quentier::IKeychainService, 56
started
quentier::SynchronizationManager, 206
StartupOption
quentier::LocalStorageManager, 106
stop
quentier::INoteStore, 79
quentier::SynchronizationManager, 206
stopped
quentier::SynchronizationManager, 207
switchUser
quentier::LocalStorageManager, 137
syncChunksDataProcessingProgress
quentier::SynchronizationManager, 207
syncChunksDownloadProgress
quentier::SynchronizationManager, 207
syncChunksDownloaded
quentier::SynchronizationManager, 207
SynchronizationManager
quentier::SynchronizationManager, 198
synchronize
quentier::SynchronizationManager, 207

tagCount
quentier::LocalStorageManager, 137
toVersion
quentier::ILocalStoragePatch, 65
totalExpungedLinkedNotebooks
quentier::ISyncChunksDataCounters, 86
totalExpungedNotebooks
quentier::ISyncChunksDataCounters, 86
totalExpungedSavedSearches
quentier::ISyncChunksDataCounters, 87
totalExpungedTags

quentier::ISyncChunksDataCounters, 87
totalLinkedNotebooks
quentier::ISyncChunksDataCounters, 87
totalNotebooks
quentier::ISyncChunksDataCounters, 87
totalSavedSearches
quentier::ISyncChunksDataCounters, 87
totalTags
quentier::ISyncChunksDataCounters, 87
type
quentier::Account, 12

undoStack
quentier::NoteEditor, 167
updateEnResource
quentier::LocalStorageManager, 138
updateLinkedNotebook
quentier::LocalStorageManager, 138
updateNote
quentier::INoteStore, 79
quentier::LocalStorageManager, 138
UpdateNoteOption
quentier::LocalStorageManager, 106
updateNotebook
quentier::INoteStore, 80
quentier::LocalStorageManager, 139
updateSavedSearch
quentier::INoteStore, 80
quentier::LocalStorageManager, 140
updateTag
quentier::INoteStore, 81
quentier::LocalStorageManager, 140
updateUser
quentier::LocalStorageManager, 141
updatedLinkedNotebooks
quentier::ISyncChunksDataCounters, 87
updatedNotebooks
quentier::ISyncChunksDataCounters, 88
updatedSavedSearches
quentier::ISyncChunksDataCounters, 88
updatedTags
quentier::ISyncChunksDataCounters, 88
upgradeProgress
quentier::LocalStorageManager, 141
userCount
quentier::LocalStorageManager, 142
userShortcut
quentier::ShortcutManager, 192, 193

value
quentier::ApplicationSettings, 20

willRepeatRemoteToLocalSyncAfterSendingChanges
quentier::SynchronizationManager, 208
writeFileRequestProcessed
quentier::FileIOProcessorAsync, 46
writePasswordJobFinished
quentier::IKeychainService, 56