

libquentier

Generated by Doxygen 1.9.4



<b>1 libquentier</b>	<b>1</b>
1.1 What's this	1
1.1.1 WARNING: libquentier is in alpha state right now, neither API nor ABI can be considered stable yet!	1
1.2 How to build/install	1
1.3 How to contribute	1
<b>2 Hierarchical Index</b>	<b>3</b>
2.1 Class Hierarchy	3
<b>3 Class Index</b>	<b>5</b>
3.1 Class List	5
<b>4 File Index</b>	<b>9</b>
4.1 File List	9
<b>5 Class Documentation</b>	<b>11</b>
5.1 quentier::Account Class Reference	11
5.1.1 Detailed Description	12
5.1.2 Member Function Documentation	13
5.1.2.1 displayName()	13
5.1.2.2 evernoteAccountType()	13
5.1.2.3 evernoteHost()	13
5.1.2.4 id()	13
5.1.2.5 isEmpty()	14
5.1.2.6 name()	14
5.1.2.7 print()	14
5.1.2.8 setDisplayName()	14
5.1.2.9 shardId()	14
5.1.2.10 type()	15
5.2 quentier::ApplicationSettings Class Reference	15
5.2.1 Detailed Description	16
5.2.2 Constructor & Destructor Documentation	16
5.2.2.1 ApplicationSettings() [1/3]	16
5.2.2.2 ApplicationSettings() [2/3]	17
5.2.2.3 ApplicationSettings() [3/3]	17
5.2.2.4 ~ApplicationSettings()	17
5.2.3 Member Function Documentation	17
5.2.3.1 beginGroup() [1/2]	18
5.2.3.2 beginGroup() [2/2]	18
5.2.3.3 beginReadArray() [1/2]	18
5.2.3.4 beginReadArray() [2/2]	19
5.2.3.5 beginWriteArray() [1/2]	19
5.2.3.6 beginWriteArray() [2/2]	19

5.2.3.7 contains() [1/2]	20
5.2.3.8 contains() [2/2]	20
5.2.3.9 print()	20
5.2.3.10 remove() [1/2]	21
5.2.3.11 remove() [2/2]	21
5.2.3.12 setValue() [1/2]	21
5.2.3.13 setValue() [2/2]	22
5.2.3.14 value() [1/2]	22
5.2.3.15 value() [2/2]	22
5.3 quantier::ApplicationSettingsInitializationException Class Reference	23
5.3.1 Detailed Description	24
5.3.2 Member Function Documentation	24
5.3.2.1 exceptionDisplayName()	24
5.4 quantier::ApplicationSettings::ArrayCloser Struct Reference	25
5.4.1 Detailed Description	25
5.5 quantier::AuthenticationManager Class Reference	25
5.5.1 Detailed Description	27
5.6 quantier::ResourceRecognitionIndexItem::BarcodeItem Struct Reference	27
5.7 quantier::DatabaseLockedException Class Reference	27
5.7.1 Member Function Documentation	28
5.7.1.1 exceptionDisplayName()	28
5.8 quantier::DatabaseLockFailedException Class Reference	29
5.8.1 Member Function Documentation	30
5.8.1.1 exceptionDisplayName()	30
5.9 quantier::DatabaseOpeningException Class Reference	30
5.9.1 Member Function Documentation	31
5.9.1.1 exceptionDisplayName()	31
5.10 quantier::DatabaseRequestException Class Reference	31
5.10.1 Detailed Description	32
5.10.2 Member Function Documentation	32
5.10.2.1 exceptionDisplayName()	32
5.11 quantier::DateTimePrint Class Reference	33
5.11.1 Detailed Description	33
5.11.2 Member Enumeration Documentation	33
5.11.2.1 Option	33
5.12 quantier::DecryptedTextManager Class Reference	33
5.13 quantier::DefaultLocalStorageCacheExpiryChecker Class Reference	34
5.13.1 Detailed Description	35
5.13.2 Member Function Documentation	35
5.13.2.1 checkLinkedNotebooks()	35
5.13.2.2 checkNotebooks()	35
5.13.2.3 checkNotes()	36

5.13.2.4 checkResources()	36
5.13.2.5 checkSavedSearches()	36
5.13.2.6 checkTags()	36
5.13.2.7 clone()	37
5.13.2.8 print()	37
5.14 quantier::EmptyDataElementException Class Reference	37
5.14.1 Member Function Documentation	38
5.14.1.1 exceptionDisplayName()	38
5.15 quantier::EncryptionManager Class Reference	39
5.15.1 Detailed Description	40
5.16 quantier::ENMLConverter Class Reference	40
5.16.1 Detailed Description	41
5.16.2 Member Function Documentation	41
5.16.2.1 cleanupExternalHtml()	41
5.16.2.2 exportNotesToEnex()	42
5.16.2.3 htmlToQTextDocument()	42
5.16.2.4 importEnex()	43
5.17 quantier::ErrorString Class Reference	43
5.17.1 Detailed Description	45
5.17.2 Member Function Documentation	45
5.17.2.1 print()	45
5.18 quantier::EventLoopWithExitStatus Class Reference	45
5.19 quantier::FileCopier Class Reference	47
5.20 quantier::FileIOProcessorAsync Class Reference	48
5.20.1 Detailed Description	49
5.20.2 Member Function Documentation	49
5.20.2.1 onReadFileRequest	49
5.20.2.2 onWriteFileRequest	50
5.20.2.3 readFileRequestProcessed	50
5.20.2.4 setIdleTimePeriod()	50
5.20.2.5 writeFileRequestProcessed	51
5.21 quantier::FileSystemWatcher Class Reference	51
5.22 quantier::ApplicationSettings::GroupCloser Struct Reference	52
5.22.1 Detailed Description	53
5.23 quantier::HTMLCleaner Class Reference	53
5.24 quantier::IAuthenticationManager Class Reference	54
5.25 quantier::IFavoritableDataElement Class Reference	55
5.25.1 Detailed Description	56
5.26 quantier::IKeychainService Class Reference	56
5.26.1 Detailed Description	57
5.26.2 Member Enumeration Documentation	57
5.26.2.1 ErrorCode	57

5.26.3 Member Function Documentation	58
5.26.3.1 deletePasswordJobFinished	58
5.26.3.2 readPasswordJobFinished	58
5.26.3.3 startDeletePasswordJob()	59
5.26.3.4 startReadPasswordJob()	59
5.26.3.5 startWritePasswordJob()	59
5.26.3.6 writePasswordJobFinished	60
5.27 quotienter::ILocalStorageCacheExpiryChecker Class Reference	60
5.27.1 Detailed Description	62
5.27.2 Member Function Documentation	62
5.27.2.1 checkLinkedNotebooks()	62
5.27.2.2 checkNotebooks()	62
5.27.2.3 checkNotes()	63
5.27.2.4 checkResources()	63
5.27.2.5 checkSavedSearches()	63
5.27.2.6 checkTags()	63
5.27.2.7 clone()	64
5.27.2.8 print()	64
5.28 quotienter::ILocalStorageDataElement Class Reference	64
5.29 quotienter::ILocalStoragePatch Class Reference	65
5.29.1 Detailed Description	66
5.29.2 Member Function Documentation	66
5.29.2.1 apply()	66
5.29.2.2 backupLocalStorage()	66
5.29.2.3 backupProgress	67
5.29.2.4 fromVersion()	67
5.29.2.5 patchLongDescription()	67
5.29.2.6 patchShortDescription()	67
5.29.2.7 progress	67
5.29.2.8 removeLocalStorageBackup()	68
5.29.2.9 restoreBackupProgress	68
5.29.2.10 restoreLocalStorageFromBackup()	68
5.29.2.11 toVersion()	69
5.30 quotienter::INoteEditorBackend Class Reference	69
5.31 quotienter::INoteStore Class Reference	72
5.31.1 Detailed Description	74
5.31.2 Member Function Documentation	74
5.31.2.1 authenticateToSharedNotebook()	74
5.31.2.2 createNote()	74
5.31.2.3 createNotebook()	75
5.31.2.4 createSavedSearch()	75
5.31.2.5 createTag()	76

5.31.2.6 getLinkedNotebookSyncChunk()	77
5.31.2.7 getLinkedNotebookSyncState()	77
5.31.2.8 getNote()	78
5.31.2.9 getNoteAsync()	79
5.31.2.10 getResource()	79
5.31.2.11 getResourceAsync()	80
5.31.2.12 getSyncChunk()	81
5.31.2.13 getSyncState()	81
5.31.2.14 noteStoreUrl()	82
5.31.2.15 setAuthData()	82
5.31.2.16 setNoteStoreUrl()	82
5.31.2.17 stop()	82
5.31.2.18 updateNote()	82
5.31.2.19 updateNotebook()	83
5.31.2.20 updateSavedSearch()	83
5.31.2.21 updateTag()	84
5.32 quantier::INoteStoreDataElement Class Reference	84
5.33 quantier::IQuantierException Class Reference	85
5.33.1 Detailed Description	87
5.33.2 Member Function Documentation	87
5.33.2.1 print()	87
5.34 quantier::ISyncStateStorage::ISyncState Class Reference	87
5.34.1 Detailed Description	88
5.34.2 Member Function Documentation	88
5.34.2.1 print()	88
5.35 quantier::ISyncStateStorage Class Reference	89
5.35.1 Detailed Description	90
5.35.2 Member Function Documentation	90
5.35.2.1 notifySyncStateUpdated	90
5.36 quantier::IUserStore Class Reference	90
5.36.1 Detailed Description	90
5.36.2 Member Function Documentation	91
5.36.2.1 checkVersion()	91
5.36.2.2 getAccountLimits()	91
5.36.2.3 getUser()	92
5.36.2.4 setAuthData()	92
5.37 quantier::LimitedStack< T > Class Template Reference	92
5.37.1 Detailed Description	93
5.38 quantier::LinkedNotebook Class Reference	94
5.38.1 Member Function Documentation	95
5.38.1.1 checkParameters()	96
5.38.1.2 clear()	96

5.38.1.3 guid()	96
5.38.1.4 hasGuid()	96
5.38.1.5 hasUpdateSequenceNumber()	96
5.38.1.6 print()	96
5.38.1.7 setGuid()	97
5.38.1.8 setUpdateSequenceNumber()	97
5.38.1.9 updateSequenceNumber()	97
5.39 quotienter::LocalStorageCacheManager Class Reference	97
5.39.1 Member Function Documentation	99
5.39.1.1 print()	99
5.40 quotienter::LocalStorageCacheManagerException Class Reference	99
5.40.1 Member Function Documentation	100
5.40.1.1 exceptionDisplayName()	100
5.41 quotienter::LocalStorageManager Class Reference	101
5.41.1 Member Enumeration Documentation	107
5.41.1.1 GetNoteOption	107
5.41.1.2 GetResourceOption	108
5.41.1.3 ListObjectsOption	108
5.41.1.4 StartupOption	108
5.41.1.5 UpdateNoteOption	109
5.41.2 Constructor & Destructor Documentation	109
5.41.2.1 LocalStorageManager()	109
5.41.3 Member Function Documentation	109
5.41.3.1 accountHighUsn()	110
5.41.3.2 addEnResource()	110
5.41.3.3 addLinkedNotebook()	110
5.41.3.4 addNote()	111
5.41.3.5 addNotebook()	111
5.41.3.6 addSavedSearch()	112
5.41.3.7 addTag()	112
5.41.3.8 addUser()	113
5.41.3.9 deleteUser()	113
5.41.3.10 enResourceCount()	113
5.41.3.11 expungeEnResource()	114
5.41.3.12 expungeLinkedNotebook()	114
5.41.3.13 expungeNote()	115
5.41.3.14 expungeNotebook()	115
5.41.3.15 expungeNotelessTagsFromLinkedNotebooks()	116
5.41.3.16 expungeSavedSearch()	116
5.41.3.17 expungeTag()	116
5.41.3.18 expungeUser()	117
5.41.3.19 findDefaultNotebook()	117



5.41.3.20 findDefaultOrLastUsedNotebook()	118
5.41.3.21 findEnResource()	118
5.41.3.22 findLastUsedNotebook()	119
5.41.3.23 findLinkedNotebook()	119
5.41.3.24 findNote()	120
5.41.3.25 findNotebook()	120
5.41.3.26 findNoteLocalUidsWithSearchQuery()	121
5.41.3.27 findNotesWithSearchQuery()	121
5.41.3.28 findSavedSearch()	121
5.41.3.29 findTag()	122
5.41.3.30 findUser()	122
5.41.3.31 highestSupportedLocalStorageVersion()	123
5.41.3.32 isLocalStorageVersionTooHigh()	123
5.41.3.33 linkedNotebookCount()	124
5.41.3.34 listAllLinkedNotebooks()	124
5.41.3.35 listAllNotebooks()	124
5.41.3.36 listAllSavedSearches()	125
5.41.3.37 listAllSharedNotebooks()	126
5.41.3.38 listAllTags()	126
5.41.3.39 listAllTagsPerNote()	127
5.41.3.40 listLinkedNotebooks()	127
5.41.3.41 listNotebooks()	128
5.41.3.42 listNotes()	129
5.41.3.43 listNotesByLocalUids()	129
5.41.3.44 listNotesPerNotebook()	130
5.41.3.45 listNotesPerNotebooksAndTags()	131
5.41.3.46 listNotesPerTag()	132
5.41.3.47 listSavedSearches()	132
5.41.3.48 listSharedNotebooksPerNotebookGuid()	133
5.41.3.49 listTags()	133
5.41.3.50 listTagsWithNoteLocalUids()	134
5.41.3.51 localStorageRequiresUpgrade()	135
5.41.3.52 localStorageVersion()	135
5.41.3.53 notebookCount()	136
5.41.3.54 noteCount()	136
5.41.3.55 noteCountPerNotebook()	136
5.41.3.56 noteCountPerNotebooksAndTags()	137
5.41.3.57 noteCountPerTag()	137
5.41.3.58 noteCountsPerAllTags()	138
5.41.3.59 requiredLocalStoragePatches()	138
5.41.3.60 savedSearchCount()	139
5.41.3.61 switchUser()	139

5.41.3.62 tagCount()	139
5.41.3.63 updateEnResource()	140
5.41.3.64 updateLinkedNotebook()	140
5.41.3.65 updateNote()	141
5.41.3.66 updateNotebook()	141
5.41.3.67 updateSavedSearch()	142
5.41.3.68 updateTag()	142
5.41.3.69 updateUser()	143
5.41.3.70 upgradeProgress	143
5.41.3.71 userCount()	144
5.42 quantier::LocalStorageManagerAsync Class Reference	144
5.43 quantier::LoggerInitializationException Class Reference	151
5.43.1 Member Function Documentation	152
5.43.1.1 exceptionDisplayName()	152
5.44 quantier::LRUCache< Key, Value, Allocator > Class Template Reference	152
5.45 quantier::Note Class Reference	153
5.45.1 Member Function Documentation	156
5.45.1.1 checkParameters()	156
5.45.1.2 clear()	156
5.45.1.3 guid()	156
5.45.1.4 hasGuid()	157
5.45.1.5 hasUpdateSequenceNumber()	157
5.45.1.6 print()	157
5.45.1.7 setGuid()	157
5.45.1.8 setUpdateSequenceNumber()	157
5.45.1.9 updateSequenceNumber()	157
5.46 quantier::Notebook Class Reference	158
5.46.1 Member Function Documentation	162
5.46.1.1 checkParameters()	162
5.46.1.2 clear()	162
5.46.1.3 guid()	162
5.46.1.4 hasGuid()	162
5.46.1.5 hasUpdateSequenceNumber()	162
5.46.1.6 print()	163
5.46.1.7 setGuid()	163
5.46.1.8 setUpdateSequenceNumber()	163
5.46.1.9 updateSequenceNumber()	163
5.47 quantier::ENMLConverter::NoteContentToHtmlExtraData Struct Reference	163
5.48 quantier::NoteEditor Class Reference	164
5.48.1 Detailed Description	167
5.48.2 Member Function Documentation	167
5.48.2.1 backend()	167

5.48.2.2 clear()	168
5.48.2.3 convertToNote	168
5.48.2.4 currentNoteLocalUid()	168
5.48.2.5 defaultFont()	168
5.48.2.6 defaultPalette()	168
5.48.2.7 idleTime()	168
5.48.2.8 inAppNoteLinkPasteRequested	169
5.48.2.9 initialize()	169
5.48.2.10 isEditorPageModified()	169
5.48.2.11 isModified()	170
5.48.2.12 isNoteLoaded()	170
5.48.2.13 saveNoteToLocalStorage	170
5.48.2.14 setAccount()	170
5.48.2.15 setBackend()	170
5.48.2.16 setCurrentNoteLocalUid()	170
5.48.2.17 setDefaultFont	171
5.48.2.18 setDefaultPalette	171
5.48.2.19 setFocus()	171
5.48.2.20 setInitialPageHtml()	172
5.48.2.21 setNoteDeletedPageHtml()	172
5.48.2.22 setNoteLoadingPageHtml()	172
5.48.2.23 setNoteNotFoundPageHtml()	172
5.48.2.24 setNoteTitle	172
5.48.2.25 setTagIds	172
5.48.2.26 setUndoStack()	173
5.48.2.27 undoStack()	173
5.49 quotient::NoteEditorInitializationException Class Reference	173
5.49.1 Member Function Documentation	174
5.49.1.1 exceptionDisplayName()	174
5.50 quotient::NoteEditorPluginInitializationException Class Reference	175
5.50.1 Member Function Documentation	176
5.50.1.1 exceptionDisplayName()	176
5.51 quotient::NoteSearchQuery Class Reference	176
5.51.1 Member Function Documentation	178
5.51.1.1 notebookModifier()	179
5.51.1.2 print()	179
5.51.1.3 queryString()	179
5.52 quotient::NullPtrException Class Reference	179
5.52.1 Member Function Documentation	180
5.52.1.1 exceptionDisplayName()	180
5.53 quotient::ResourceRecognitionIndexItem::ObjectItem Struct Reference	180
5.54 quotient::Printable Class Reference	181

5.54.1 Detailed Description	182
5.54.2 Member Function Documentation	182
5.54.2.1 print()	182
5.55 quantier::QuantierApplication Class Reference	182
5.56 quantier::QuantierUndoCommand Class Reference	183
5.56.1 Detailed Description	184
5.57 quantier::Resource Class Reference	185
5.57.1 Member Function Documentation	187
5.57.1.1 checkParameters()	187
5.57.1.2 clear()	187
5.57.1.3 guid()	187
5.57.1.4 hasGuid()	188
5.57.1.5 hasUpdateSequenceNumber()	188
5.57.1.6 print()	188
5.57.1.7 setGuid()	188
5.57.1.8 setUpdateSequenceNumber()	188
5.57.1.9 updateSequenceNumber()	188
5.58 quantier::ResourceRecognitionIndexItem Class Reference	189
5.58.1 Member Function Documentation	191
5.58.1.1 print()	191
5.59 quantier::ResourceRecognitionIndices Class Reference	191
5.59.1 Member Function Documentation	192
5.59.1.1 print()	192
5.60 quantier::SavedSearch Class Reference	193
5.60.1 Member Function Documentation	195
5.60.1.1 checkParameters()	195
5.60.1.2 clear()	195
5.60.1.3 guid()	196
5.60.1.4 hasGuid()	196
5.60.1.5 hasUpdateSequenceNumber()	196
5.60.1.6 print()	196
5.60.1.7 setGuid()	196
5.60.1.8 setUpdateSequenceNumber()	196
5.60.1.9 updateSequenceNumber()	197
5.61 quantier::ResourceRecognitionIndexItem::ShapeItem Struct Reference	197
5.62 quantier::SharedNote Class Reference	197
5.62.1 Member Function Documentation	199
5.62.1.1 print()	199
5.63 quantier::SharedNotebook Class Reference	200
5.63.1 Member Function Documentation	202
5.63.1.1 print()	202
5.64 quantier::ShortcutManager Class Reference	202

5.64.1 Member Function Documentation	203
5.64.1.1 defaultShortcut() [1/2]	204
5.64.1.2 defaultShortcut() [2/2]	204
5.64.1.3 shortcut() [1/2]	204
5.64.1.4 shortcut() [2/2]	204
5.64.1.5 userShortcut() [1/2]	205
5.64.1.6 userShortcut() [2/2]	205
5.65 quotienter::ENMLConverter::SkipHtmlElementRule Class Reference	205
5.65.1 Detailed Description	206
5.65.2 Member Function Documentation	207
5.65.2.1 print()	207
5.66 quotienter::SpellChecker Class Reference	207
5.67 quotienter::StringUtils::StringFilterPredicate Struct Reference	208
5.68 quotienter::StringUtils Class Reference	208
5.69 quotienter::SynchronizationManager Class Reference	209
5.69.1 Detailed Description	210
5.69.2 Constructor & Destructor Documentation	210
5.69.2.1 SynchronizationManager()	211
5.69.3 Member Function Documentation	211
5.69.3.1 active()	211
5.69.3.2 authenticate	212
5.69.3.3 authenticateCurrentAccount	212
5.69.3.4 authenticationFinished	212
5.69.3.5 authenticationRevoked	212
5.69.3.6 detectedConflictDuringLocalChangesSending	213
5.69.3.7 downloadNoteThumbnailsOption()	213
5.69.3.8 failed	213
5.69.3.9 finished	213
5.69.3.10 linkedNotebooksNotesDownloadProgress	214
5.69.3.11 linkedNotebooksResourcesDownloadProgress	214
5.69.3.12 linkedNotebooksSyncChunksDownloaded	214
5.69.3.13 linkedNotebookSyncChunksDownloadProgress	215
5.69.3.14 notesDownloadProgress	215
5.69.3.15 preparedDirtyObjectsForSending	215
5.69.3.16 preparedLinkedNotebooksDirtyObjectsForSending	215
5.69.3.17 rateLimitExceeded	216
5.69.3.18 remoteToLocalSyncDone	216
5.69.3.19 remoteToLocalSyncStopped	216
5.69.3.20 resourcesDownloadProgress	216
5.69.3.21 revokeAuthentication	217
5.69.3.22 sendLocalChangesStopped	217
5.69.3.23 setAccount	217

5.69.3.24 setAccountDone . . . . .	217
5.69.3.25 setDownloadInkNotelImages . . . . .	217
5.69.3.26 setDownloadInkNotelImagesDone . . . . .	218
5.69.3.27 setDownloadNoteThumbnails . . . . .	218
5.69.3.28 setDownloadNoteThumbnailsDone . . . . .	218
5.69.3.29 setInkNotelImagesStoragePath . . . . .	218
5.69.3.30 setInkNotelImagesStoragePathDone . . . . .	218
5.69.3.31 started . . . . .	219
5.69.3.32 stop . . . . .	219
5.69.3.33 stopped . . . . .	219
5.69.3.34 syncChunksDownloaded . . . . .	219
5.69.3.35 syncChunksDownloadProgress . . . . .	219
5.69.3.36 synchronize . . . . .	220
5.69.3.37 willRepeatRemoteToLocalSyncAfterSendingChanges . . . . .	220
5.70 quotienter::SysInfo Class Reference . . . . .	220
5.71 quotienter::Tag Class Reference . . . . .	220
5.71.1 Member Function Documentation . . . . .	222
5.71.1.1 checkParameters() . . . . .	222
5.71.1.2 clear() . . . . .	222
5.71.1.3 guid() . . . . .	222
5.71.1.4 hasGuid() . . . . .	222
5.71.1.5 hasUpdateSequenceNumber() . . . . .	223
5.71.1.6 print() . . . . .	223
5.71.1.7 setGuid() . . . . .	223
5.71.1.8 setUpdateSequenceNumber() . . . . .	223
5.71.1.9 updateSequenceNumber() . . . . .	223
5.72 quotienter::ResourceRecognitionIndexItem::TextItem Struct Reference . . . . .	223
5.73 quotienter::UidGenerator Class Reference . . . . .	224
5.74 quotienter::User Class Reference . . . . .	224
5.74.1 Member Function Documentation . . . . .	226
5.74.1.1 print() . . . . .	226
<b>6 File Documentation</b>	<b>227</b>
6.1 DecryptedTextManager.h . . . . .	227
6.2 ENMLConverter.h . . . . .	228
6.3 HTMLCleaner.h . . . . .	230
6.4 ApplicationSettingsInitializationException.h . . . . .	230
6.5 DatabaseLockedException.h . . . . .	231
6.6 DatabaseLockFailedException.h . . . . .	231
6.7 DatabaseOpeningException.h . . . . .	232
6.8 DatabaseRequestException.h . . . . .	232
6.9 EmptyDataElementException.h . . . . .	233

6.10 IQuentierException.h . . . . .	233
6.11 LocalStorageCacheManagerException.h . . . . .	234
6.12 LoggerInitializationException.h . . . . .	235
6.13 NoteEditorInitializationException.h . . . . .	235
6.14 NoteEditorPluginInitializationException.h . . . . .	236
6.15 NullPtrException.h . . . . .	236
6.16 DefaultLocalStorageCacheExpiryChecker.h . . . . .	237
6.17 ILocalStorageCacheExpiryChecker.h . . . . .	237
6.18 ILocalStoragePatch.h . . . . .	238
6.19 Lists.h . . . . .	239
6.20 LocalStorageCacheManager.h . . . . .	240
6.21 LocalStorageManager.h . . . . .	241
6.22 LocalStorageManagerAsync.h . . . . .	248
6.23 NoteSearchQuery.h . . . . .	257
6.24 QuentierLogger.h . . . . .	259
6.25 INoteEditorBackend.h . . . . .	260
6.26 NoteEditor.h . . . . .	263
6.27 SpellChecker.h . . . . .	266
6.28 AuthenticationManager.h . . . . .	267
6.29 synchronization/ForwardDeclarations.h . . . . .	268
6.30 utility/ForwardDeclarations.h . . . . .	268
6.31 IAuthenticationManager.h . . . . .	269
6.32 INoteStore.h . . . . .	270
6.33 ISyncStateStorage.h . . . . .	272
6.34 IUserStore.h . . . . .	272
6.35 SynchronizationManager.h . . . . .	273
6.36 Account.h . . . . .	275
6.37 ErrorString.h . . . . .	277
6.38 IFavoritableDataElement.h . . . . .	277
6.39 ILocalStorageDataElement.h . . . . .	278
6.40 INoteStoreDataElement.h . . . . .	279
6.41 LinkedNotebook.h . . . . .	281
6.42 Note.h . . . . .	282
6.43 Notebook.h . . . . .	285
6.44 RegisterMetatypes.h . . . . .	288
6.45 Resource.h . . . . .	288
6.46 ResourceRecognitionIndexItem.h . . . . .	290
6.47 ResourceRecognitionIndices.h . . . . .	292
6.48 SavedSearch.h . . . . .	293
6.49 SharedNote.h . . . . .	294
6.50 SharedNotebook.h . . . . .	296
6.51 Tag.h . . . . .	298

6.52 User.h . . . . .	299
6.53 ApplicationSettings.h . . . . .	301
6.54 Checks.h . . . . .	302
6.55 Compat.h . . . . .	303
6.56 DateTime.h . . . . .	303
6.57 EncryptionManager.h . . . . .	304
6.58 EventLoopWithExitStatus.h . . . . .	305
6.59 FileCopier.h . . . . .	306
6.60 FileIOProcessorAsync.h . . . . .	307
6.61 FileSystem.h . . . . .	308
6.62 FileSystemWatcher.h . . . . .	308
6.63 IKeychainService.h . . . . .	309
6.64 Initialize.h . . . . .	311
6.65 LimitedStack.h . . . . .	311
6.66 Linkage.h . . . . .	312
6.67 LRUCache.hpp . . . . .	313
6.68 MessageBox.h . . . . .	316
6.69 Printable.h . . . . .	316
6.70 QuentierApplication.h . . . . .	318
6.71 QuentierCheckPtr.h . . . . .	318
6.72 QuentierUndoCommand.h . . . . .	319
6.73 ShortcutManager.h . . . . .	319
6.74 Size.h . . . . .	322
6.75 StandardPaths.h . . . . .	322
6.76 StringUtils.h . . . . .	323
6.77 SuppressWarnings.h . . . . .	323
6.78 SysInfo.h . . . . .	325
6.79 System.h . . . . .	325
6.80 TagSortByParentChildRelations.h . . . . .	326
6.81 UidGenerator.h . . . . .	326
<b>Index</b>	<b>329</b>



# Chapter 1

## libquentier

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

### 1.1 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.

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

### 1.2 How to build/install

Please see the building/installation guide.

### 1.3 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 . . . . .	25
quentier::ResourceRecognitionIndexItem::BarcodeItem . . . . .	27
quentier::DateTimePrint . . . . .	33
quentier::DecryptedTextManager . . . . .	33
quentier::ENMLConverter . . . . .	40
std::exception	
quentier::IQuentierException . . . . .	85
quentier::ApplicationSettingsInitializationException . . . . .	23
quentier::DatabaseLockFailedException . . . . .	29
quentier::DatabaseLockedException . . . . .	27
quentier::DatabaseOpeningException . . . . .	30
quentier::DatabaseRequestException . . . . .	31
quentier::EmptyDataElementException . . . . .	37
quentier::LocalStorageCacheManagerException . . . . .	99
quentier::LoggerInitializationException . . . . .	151
quentier::NoteEditorInitializationException . . . . .	173
quentier::NoteEditorPluginInitializationException . . . . .	175
quentier::NullPtrException . . . . .	179
quentier::ApplicationSettings::GroupCloser . . . . .	52
quentier::HTMLCleaner . . . . .	53
quentier::ILocalStorageDataElement . . . . .	64
quentier::INoteStoreDataElement . . . . .	84
quentier::IFavoritableDataElement . . . . .	55
quentier::Note . . . . .	153
quentier::Notebook . . . . .	158
quentier::SavedSearch . . . . .	193
quentier::Tag . . . . .	220
quentier::LinkedNotebook . . . . .	94
quentier::Resource . . . . .	185
quentier::INoteEditorBackend . . . . .	69
quentier::IUserStore . . . . .	90
quentier::LRUCache< Key, Value, Allocator > . . . . .	152
quentier::ENMLConverter::NoteContentToHtmlExtraData . . . . .	163
quentier::ResourceRecognitionIndexItem::ObjectItem . . . . .	180
quentier::Printable . . . . .	181

quentier::Account	11
quentier::ApplicationSettings	15
quentier::ENMLConverter::SkipHtmlElementRule	205
quentier::ErrorString	43
quentier::ILocalStorageCacheExpiryChecker	60
quentier::DefaultLocalStorageCacheExpiryChecker	34
quentier::INoteStoreDataElement	84
quentier::IQuentierException	85
quentier::ISyncStateStorage::ISyncState	87
quentier::LocalStorageCacheManager	97
quentier::NoteSearchQuery	176
quentier::ResourceRecognitionIndexItem	189
quentier::ResourceRecognitionIndices	191
quentier::SharedNote	197
quentier::SharedNotebook	200
quentier::User	224
QApplication	
quentier::QuentierApplication	182
QEventLoop	
quentier::EventLoopWithExitStatus	45
QObject	
quentier::EncryptionManager	39
quentier::FileCopier	47
quentier::FileIOProcessorAsync	48
quentier::FileSystemWatcher	51
quentier::IAuthenticationManager	54
quentier::AuthenticationManager	25
quentier::IKeychainService	56
quentier::ILocalStoragePatch	65
quentier::INoteStore	72
quentier::ISyncStateStorage	89
quentier::LocalStorageManager	101
quentier::LocalStorageManagerAsync	144
quentier::QuentierUndoCommand	183
quentier::ShortcutManager	202
quentier::SpellChecker	207
quentier::SynchronizationManager	209
QSettings	
quentier::ApplicationSettings	15
QStack	
quentier::LimitedStack< T >	92
QUndoCommand	
quentier::QuentierUndoCommand	183
QWidget	
quentier::NoteEditor	164
quentier::ResourceRecognitionIndexItem::ShapeItem	197
quentier::StringUtils::StringFilterPredicate	208
quentier::StringUtils	208
quentier::SysInfo	220
quentier::ResourceRecognitionIndexItem::TextItem	223
quentier::UidGenerator	224

## Chapter 3

# Class Index

### 3.1 Class List

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

<a href="#">quentier::Account</a>	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 . . . . .	11
<a href="#">quentier::ApplicationSettings</a>	Enhances the functionality of QSettings, in particular it simplifies the way of working with either application-wide or account-specific settings . . . . .	15
<a href="#">quentier::ApplicationSettingsInitializationException</a>	The <a href="#">ApplicationSettingsInitializationException</a> can be thrown from methods of <a href="#">ApplicationSettings</a> class if it's unable to locate the file with persistent settings . . . . .	23
<a href="#">quentier::ApplicationSettings::ArrayCloser</a>		25
<a href="#">quentier::AuthenticationManager</a>	Libquentier's default implementation of <a href="#">IAuthenticationManager</a> interface; internally uses QEverCloud's OAuth widget . . . . .	25
<a href="#">quentier::ResourceRecognitionIndexItem::BarcodeItem</a>		27
<a href="#">quentier::DatabaseLockedException</a>		27
<a href="#">quentier::DatabaseLockFailedException</a>		29
<a href="#">quentier::DatabaseOpeningException</a>		30
<a href="#">quentier::DatabaseRequestException</a>	The <a href="#">DatabaseRequestException</a> is thrown when the local storage database encounters some internal error during the attempt to serve a request . . . . .	31
<a href="#">quentier::DateTimePrint</a>	Simply wraps the enum containing datetime printing options . . . . .	33
<a href="#">quentier::DecryptedTextManager</a>		33
<a href="#">quentier::DefaultLocalStorageCacheExpiryChecker</a>		34
<a href="#">quentier::EmptyDataElementException</a>		37
<a href="#">quentier::EncryptionManager</a>	Both synchronous methods to encrypt or decrypt given text with password, cipher and key length and their signal-slot based potentially asynchronous counterparts . . . . .	39
<a href="#">quentier::ENMLConverter</a>	Encapsulates a set of methods and helper data structures for performing the conversions between ENML and other note content formats, namely HTML . . . . .	40
<a href="#">quentier::ErrorString</a>	Encapsulates two (or more) strings which are meant to contain translatable (base) and non-translatable (details) parts of the error description . . . . .	43
<a href="#">quentier::EventLoopWithExitStatus</a>		45

<a href="#">quentier::FileCopier</a>	47
<a href="#">quentier::FileIOProcessorAsync</a>	
Wrapper under simple file IO operations, it is meant to be used for simple asynchronous IO	48
<a href="#">quentier::FileSystemWatcher</a>	51
<a href="#">quentier::ApplicationSettings::GroupCloser</a>	52
<a href="#">quentier::HTMLCleaner</a>	53
<a href="#">quentier::IAuthenticationManager</a>	54
<a href="#">quentier::IFavoritableDataElement</a>	55
<a href="#">quentier::IKeychainService</a>	
The <a href="#">IKeychainService</a> interface provides methods intended to start potentially asynchronous interaction with the keychain and signals intended to notify listeners about the completion of asynchronous interactions	56
<a href="#">quentier::ILocalStorageCacheExpiryChecker</a>	
Interface for cache expiry checker used by <a href="#">LocalStorageCacheManager</a> to see whether particular caches (of notes, notebooks, tags, linked notebooks and/or saved searches) need to be shrunk	60
<a href="#">quentier::ILocalStorageDataElement</a>	64
<a href="#">quentier::ILocalStoragePatch</a>	
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	65
<a href="#">quentier::INoteEditorBackend</a>	69
<a href="#">quentier::INoteStore</a>	
<a href="#">INoteStore</a> is the interface which provides methods required for the implementation of NoteStore part of Evernote EDAM sync protocol	72
<a href="#">quentier::INoteStoreDataElement</a>	84
<a href="#">quentier::IQuentierException</a>	
Interface for exceptions specific to libquentier and applications based on it	85
<a href="#">quentier::ISyncStateStorage::ISyncState</a>	
The <a href="#">ISyncState</a> interface provides accessory methods to determine the sync state for the account	87
<a href="#">quentier::ISyncStateStorage</a>	
The <a href="#">ISyncStateStorage</a> interface represents the interface of a class which stores sync state for given accounts persistently and provides access to previously stores sync states	89
<a href="#">quentier::IUserStore</a>	
<a href="#">IUserStore</a> is the interface which provides methods required for the implementation of UserStore part of Evernote EDAM sync protocol	90
<a href="#">quentier::LimitedStack&lt; T &gt;</a>	
The <a href="#">LimitedStack</a> 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	92
<a href="#">quentier::LinkedNotebook</a>	94
<a href="#">quentier::LocalStorageCacheManager</a>	97
<a href="#">quentier::LocalStorageCacheManagerException</a>	99
<a href="#">quentier::LocalStorageManager</a>	101
<a href="#">quentier::LocalStorageManagerAsync</a>	144
<a href="#">quentier::LoggerInitializationException</a>	151
<a href="#">quentier::LRUCache&lt; Key, Value, Allocator &gt;</a>	152
<a href="#">quentier::Note</a>	153
<a href="#">quentier::Notebook</a>	158
<a href="#">quentier::ENMLConverter::NoteContentToHtmlExtraData</a>	163
<a href="#">quentier::NoteEditor</a>	
Widget encapsulating all the functionality necessary for showing and editing notes	164
<a href="#">quentier::NoteEditorInitializationException</a>	173
<a href="#">quentier::NoteEditorPluginInitializationException</a>	175
<a href="#">quentier::NoteSearchQuery</a>	176
<a href="#">quentier::NullPtrException</a>	179
<a href="#">quentier::ResourceRecognitionIndexItem::ObjectItem</a>	180

<a href="#">quentier::Printable</a>	
Interface for Quentier's internal classes which should be able to write themselves into QText↔ Stream and/or convert to QString . . . . .	181
<a href="#">quentier::QuentierApplication</a> . . . . .	182
<a href="#">quentier::QuentierUndoCommand</a>	
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 . . .	183
<a href="#">quentier::Resource</a> . . . . .	185
<a href="#">quentier::ResourceRecognitionIndexItem</a> . . . . .	189
<a href="#">quentier::ResourceRecognitionIndices</a> . . . . .	191
<a href="#">quentier::SavedSearch</a> . . . . .	193
<a href="#">quentier::ResourceRecognitionIndexItem::ShapelItem</a> . . . . .	197
<a href="#">quentier::SharedNote</a> . . . . .	197
<a href="#">quentier::SharedNotebook</a> . . . . .	200
<a href="#">quentier::ShortcutManager</a> . . . . .	202
<a href="#">quentier::ENMLConverter::SkipHtmlElementRule</a>	
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 . . . . .	205
<a href="#">quentier::SpellChecker</a> . . . . .	207
<a href="#">quentier::StringUtils::StringFilterPredicate</a> . . . . .	208
<a href="#">quentier::StringUtils</a> . . . . .	208
<a href="#">quentier::SynchronizationManager</a>	
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 . . . . .	209
<a href="#">quentier::SysInfo</a> . . . . .	220
<a href="#">quentier::Tag</a> . . . . .	220
<a href="#">quentier::ResourceRecognitionIndexItem::TextItem</a> . . . . .	223
<a href="#">quentier::UidGenerator</a> . . . . .	224
<a href="#">quentier::User</a> . . . . .	224





## Chapter 4

# File Index

### 4.1 File List

Here is a list of all documented files with brief descriptions:

<a href="#">DecryptedTextManager.h</a>	227
<a href="#">ENMLConverter.h</a>	228
<a href="#">HTMLCleaner.h</a>	230
<a href="#">ApplicationSettingsInitializationException.h</a>	230
<a href="#">DatabaseLockedException.h</a>	231
<a href="#">DatabaseLockFailedException.h</a>	231
<a href="#">DatabaseOpeningException.h</a>	232
<a href="#">DatabaseRequestException.h</a>	232
<a href="#">EmptyDataElementException.h</a>	233
<a href="#">IQuentierException.h</a>	233
<a href="#">LocalStorageCacheManagerException.h</a>	234
<a href="#">LoggerInitializationException.h</a>	235
<a href="#">NoteEditorInitializationException.h</a>	235
<a href="#">NoteEditorPluginInitializationException.h</a>	236
<a href="#">NullPtrException.h</a>	236
<a href="#">DefaultLocalStorageCacheExpiryChecker.h</a>	237
<a href="#">ILocalStorageCacheExpiryChecker.h</a>	237
<a href="#">ILocalStoragePatch.h</a>	238
<a href="#">Lists.h</a>	239
<a href="#">LocalStorageCacheManager.h</a>	240
<a href="#">LocalStorageManager.h</a>	241
<a href="#">LocalStorageManagerAsync.h</a>	248
<a href="#">NoteSearchQuery.h</a>	257
<a href="#">QuentierLogger.h</a>	259
<a href="#">INoteEditorBackend.h</a>	260
<a href="#">NoteEditor.h</a>	263
<a href="#">SpellChecker.h</a>	266
<a href="#">AuthenticationManager.h</a>	267
<a href="#">synchronization/ForwardDeclarations.h</a>	268
<a href="#">utility/ForwardDeclarations.h</a>	268
<a href="#">IAuthenticationManager.h</a>	269
<a href="#">INoteStore.h</a>	270
<a href="#">ISyncStateStorage.h</a>	272
<a href="#">IUserStore.h</a>	272
<a href="#">SynchronizationManager.h</a>	273

Account.h	275
ErrorString.h	277
IFavoritableDataElement.h	277
ILocalStorageDataElement.h	278
INoteStoreDataElement.h	279
LinkedNotebook.h	281
Note.h	282
Notebook.h	285
RegisterMetatypes.h	288
Resource.h	288
ResourceRecognitionIndexItem.h	290
ResourceRecognitionIndices.h	292
SavedSearch.h	293
SharedNote.h	294
SharedNotebook.h	296
Tag.h	298
User.h	299
ApplicationSettings.h	301
Checks.h	302
Compat.h	303
DateTime.h	303
EncryptionManager.h	304
EventLoopWithExitStatus.h	305
FileCopier.h	306
FileIOProcessorAsync.h	307
FileSystem.h	308
FileSystemWatcher.h	308
IKeychainService.h	309
Initialize.h	311
LimitedStack.h	311
Linkage.h	312
LRUCache.hpp	313
MessageBox.h	316
Printable.h	316
QuentierApplication.h	318
QuentierCheckPtr.h	318
QuentierUndoCommand.h	319
ShortcutManager.h	319
Size.h	322
StandardPaths.h	322
StringUtils.h	323
SuppressWarnings.h	323
SysInfo.h	325
System.h	325
TagSortByParentChildRelations.h	326
UidGenerator.h	326

## Chapter 5

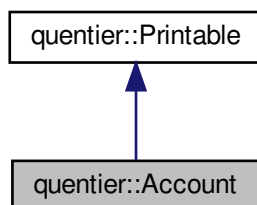
# Class Documentation

### 5.1 `quentier::Account` Class Reference

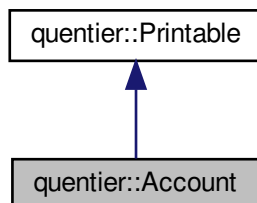
The [Account](#) class encapsulates some details about the account: its name, whether it is local or synchronized to Evernote and for the latter case - some additional details like upload limit etc.

```
#include <Account.h>
```

Inheritance diagram for `quentier::Account`:



Collaboration diagram for `quentier::Account`:



## Public Types

- enum class **Type** { **Local** = 0 , **Evernote** }
- enum class **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

### 5.1.1 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.

## 5.1.2 Member Function Documentation

### 5.1.2.1 displayName()

```
QString quantier::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

### 5.1.2.2 evernoteAccountType()

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

#### Returns

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

### 5.1.2.3 evernoteHost()

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

#### Returns

The Evernote server host with which the account is associated

### 5.1.2.4 id()

```
qevercloud::UserID quantier::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)

#### 5.1.2.5 isEmpty()

```
bool quantier::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

#### 5.1.2.6 name()

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

##### Returns

Username for either local or Evernote account

#### 5.1.2.7 print()

```
virtual QTextStream & quantier::Account::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [quantier::Printable](#).

#### 5.1.2.8 setDisplayName()

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

Set the printable name of the account

#### 5.1.2.9 shardId()

```
QString quantier::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)

### 5.1.2.10 type()

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

#### Returns

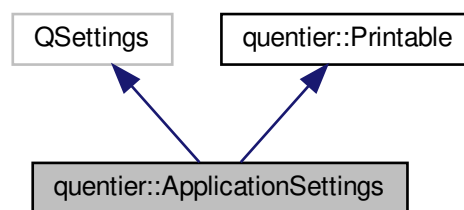
The type of the account: either local of Evernote

## 5.2 quantier::ApplicationSettings Class Reference

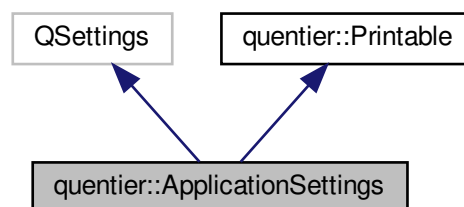
The [ApplicationSettings](#) class enhances the functionality of QSettings, in particular it simplifies the way of working with either application-wide or account-specific settings.

```
#include <ApplicationSettings.h>
```

Inheritance diagram for quantier::ApplicationSettings:



Collaboration diagram for quantier::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

### 5.2.1 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.

### 5.2.2 Constructor & Destructor Documentation

#### 5.2.2.1 ApplicationSettings() [1/3]

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

Constructor for application settings not being account-specific

#### Parameters

<i>settingsName</i>	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
---------------------	--



### 5.2.2.2 ApplicationSettings() [2/3]

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

Constructor for application settings specific to the account

#### Parameters

<i>account</i>	The account for which the settings are to be stored or read
<i>settingsName</i>	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

### 5.2.2.3 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

<i>account</i>	The account for which the settings are to be stored or read
<i>settingsName</i>	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
<i>settingsNameSize</i>	Size of the settingsName string. If negative (the default), the settingsName size is taken to be strlen(settingsName)

### 5.2.2.4 ~ApplicationSettings()

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

Destructor

## 5.2.3 Member Function Documentation

### 5.2.3.1 beginGroup() [1/2]

```
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

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

### 5.2.3.2 beginGroup() [2/2]

```
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

<i>prefix</i>	String containing the prefix name
---------------	-----------------------------------

### 5.2.3.3 beginReadArray() [1/2]

```
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

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

**5.2.3.4 beginReadArray() [2/2]**

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

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

**Parameters**

<i>prefix</i>	String containing the prefix name
---------------	-----------------------------------

**Returns**

The size of the array

**5.2.3.5 beginWriteArray() [1/2]**

```
void quotientier::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**

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

**5.2.3.6 beginWriteArray() [2/2]**

```
void quotientier::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**

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

### 5.2.3.7 contains() [1/2]

```
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

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

#### Returns

True if there exists a setting called key; false otherwise

### 5.2.3.8 contains() [2/2]

```
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

<i>key</i>	The key being checked for presence
------------	------------------------------------

#### Returns

True if there exists a setting called key; false otherwise

### 5.2.3.9 print()

```
virtual QTextStream & quentier::ApplicationSettings::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [quentier::Printable](#).

**5.2.3.10 remove()** [1/2]

```
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**

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

**5.2.3.11 remove()** [2/2]

```
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**

<i>key</i>	String containing the setting name
------------	------------------------------------

**5.2.3.12 setValue()** [1/2]

```
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**

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

**5.2.3.13 setValue() [2/2]**

```
void quantier::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**

<i>key</i>	String containing the setting name
<i>value</i>	Value for setting key

**5.2.3.14 value() [1/2]**

```
QVariant quantier::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**

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

**5.2.3.15 value() [2/2]**

```
QVariant quantier::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

<i>key</i>	String containing the setting name
<i>defaultValue</i>	Default value returned if the setting doesn't exist

## Returns

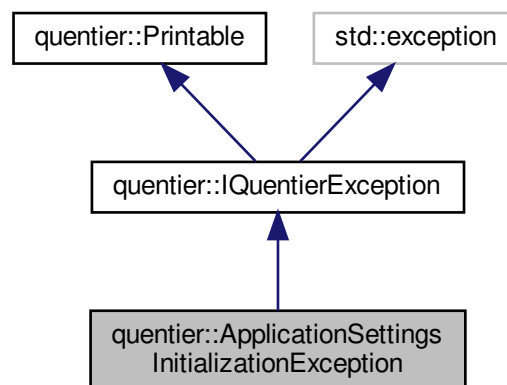
The value for setting `key`. If the setting doesn't exist, returns `defaultValue`. If no default value is specified, a default `QVariant` is returned.

## 5.3 `quentier::ApplicationSettingsInitializationException` Class Reference

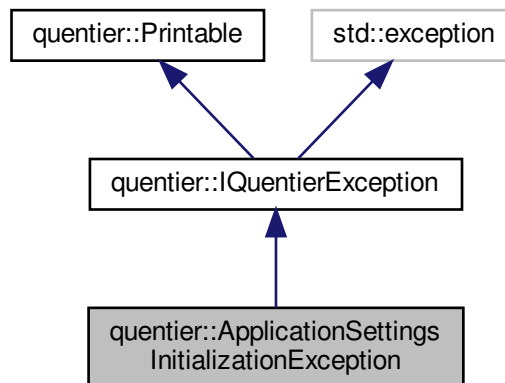
The [ApplicationSettingsInitializationException](#) can be thrown from methods of [ApplicationSettings](#) class if it's unable to locate the file with persistent settings.

```
#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

### 5.3.1 Detailed Description

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

### 5.3.2 Member Function Documentation

#### 5.3.2.1 exceptionDisplayName()

```
virtual const QString quentier::ApplicationSettingsInitializationException::exceptionDisplay←
Name ( ) const [override], [protected], [virtual]
```

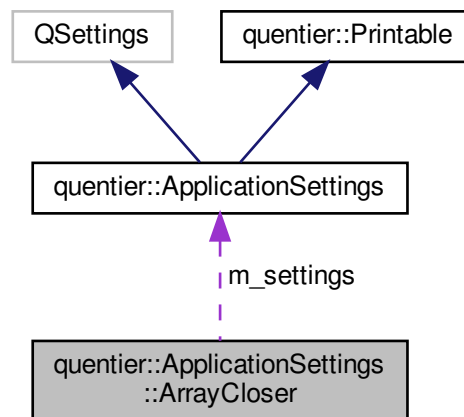
Implements [quentier::IQuentierException](#).



## 5.4 `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**

#### 5.4.1 Detailed Description

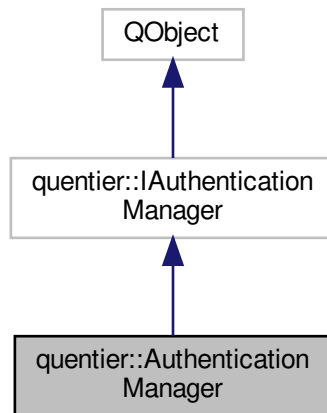
Helper struct for RAII style of ensuring the array once began would be closed even if exception is thrown after beginning the array

## 5.5 `quentier::AuthenticationManager` Class Reference

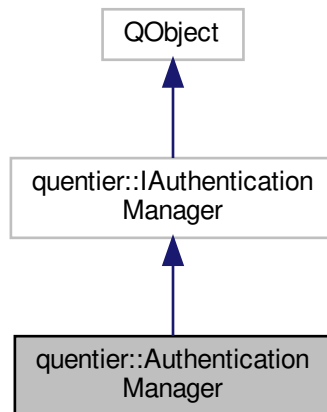
The [AuthenticationManager](#) class is libquentier's default implementation of [IAuthenticationManager](#) interface; internally uses QEverCloud's OAuth widget.

```
#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

### 5.5.1 Detailed Description

The [AuthenticationManager](#) class is libquentier's default implementation of [IAuthenticationManager](#) interface; internally uses QEverCloud's OAuth widget.

## 5.6 `quentier::ResourceRecognitionIndexItem::BarcodeItem` Struct Reference

### Public Member Functions

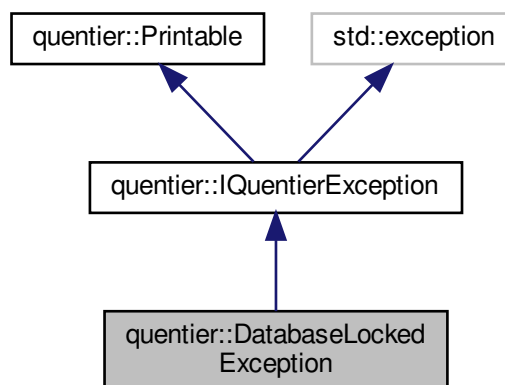
- `bool operator== (const BarcodeItem &other) const`

### Public Attributes

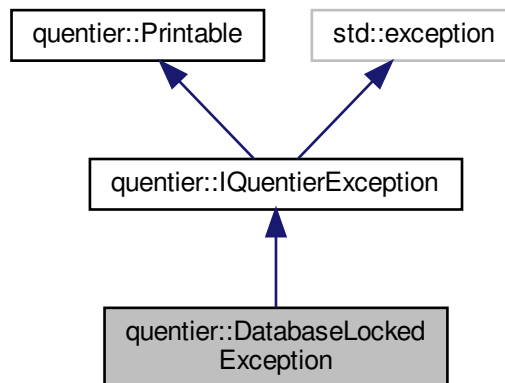
- `QString m_barcode`
- `int m_weight = -1`

## 5.7 `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

## 5.7.1 Member Function Documentation

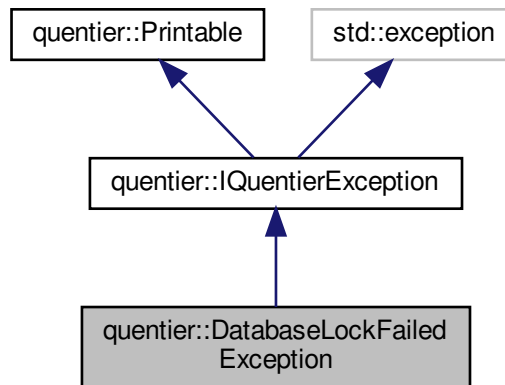
### 5.7.1.1 exceptionDisplayName()

```
virtual const QString quentier::DatabaseLockedException::exceptionDisplayName ( ) const [override],  
[protected], [virtual]
```

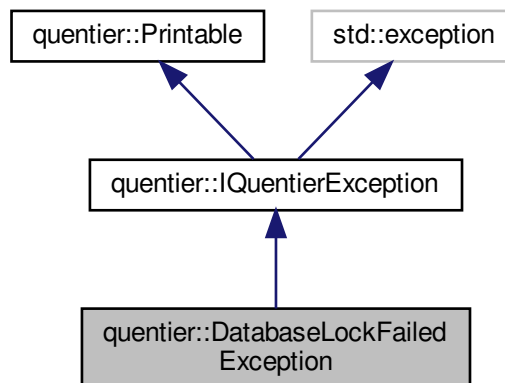
Implements [quentier::IQuentierException](#).

## 5.8 quantier::DatabaseLockFailedException Class Reference

Inheritance diagram for quantier::DatabaseLockFailedException:



Collaboration diagram for quantier::DatabaseLockFailedException:



### Public Member Functions

- **DatabaseLockFailedException** (const [ErrorString](#) &message)

### Protected Member Functions

- virtual const QString [exceptionDisplayName](#) () const override

## 5.8.1 Member Function Documentation

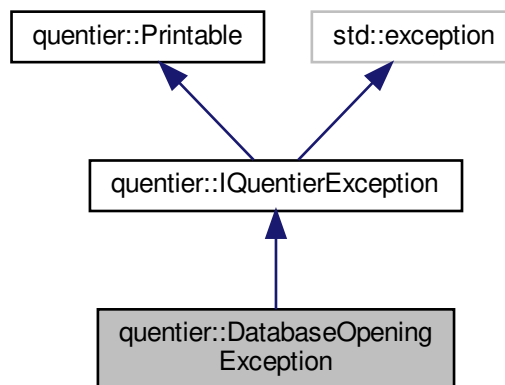
### 5.8.1.1 exceptionDisplayName()

```
virtual const QString quantier::DatabaseLockFailedException::exceptionDisplayName ( ) const  
[override], [protected], [virtual]
```

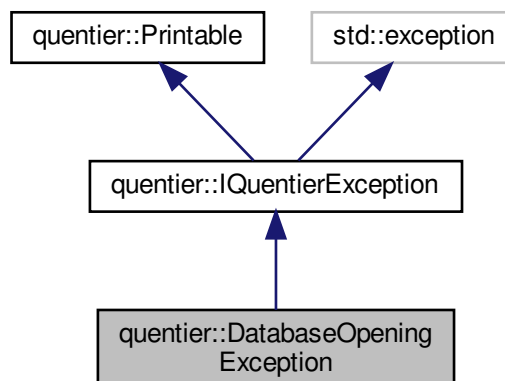
Implements [quantier::IQuantierException](#).

## 5.9 quantier::DatabaseOpeningException Class Reference

Inheritance diagram for quantier::DatabaseOpeningException:



Collaboration diagram for quantier::DatabaseOpeningException:



## Public Member Functions

- **DatabaseOpeningException** (const [ErrorString](#) &message)

## Protected Member Functions

- virtual const QString [exceptionDisplayName](#) () const override

### 5.9.1 Member Function Documentation

#### 5.9.1.1 exceptionDisplayName()

```
virtual const QString quantier::DatabaseOpeningException::exceptionDisplayName ( ) const [override],  
[protected], [virtual]
```

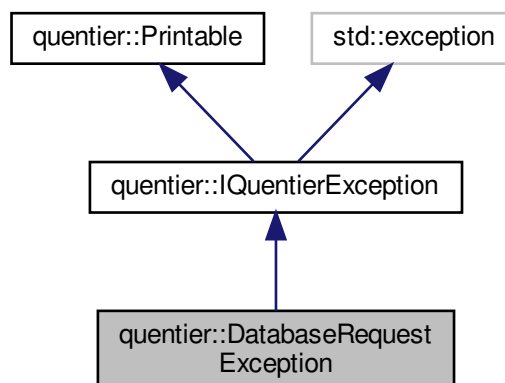
Implements [quantier::IQuantierException](#).

## 5.10 quantier::DatabaseRequestException Class Reference

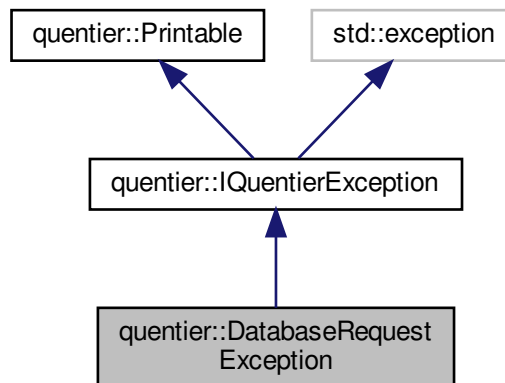
The [DatabaseRequestException](#) is thrown when the local storage database encounters some internal error during the attempt to serve a request.

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

Inheritance diagram for quantier::DatabaseRequestException:



Collaboration diagram for `quentier::DatabaseRequestException`:



## Public Member Functions

- **DatabaseRequestException** (const [ErrorString](#) &message)

## Protected Member Functions

- virtual const QString [exceptionDisplayName](#) () const override

### 5.10.1 Detailed Description

The [DatabaseRequestException](#) is thrown when the local storage database encounters some internal error during the attempt to serve a request.

### 5.10.2 Member Function Documentation

#### 5.10.2.1 exceptionDisplayName()

```
virtual const QString quentier::DatabaseRequestException::exceptionDisplayName ( ) const [override],  
[protected], [virtual]
```

Implements [quentier::IQuentierException](#).



## 5.11 quantier::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 }

#### 5.11.1 Detailed Description

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

#### 5.11.2 Member Enumeration Documentation

##### 5.11.2.1 Option

```
enum quantier::DateTimePrint::Option
```

Available printing options for datetime

##### Enumerator

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

## 5.12 quantier::DecryptedTextManager Class Reference

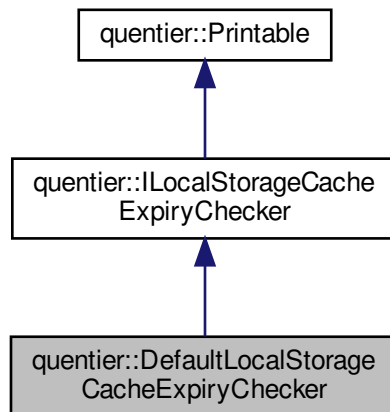
### Public Member Functions

- 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)

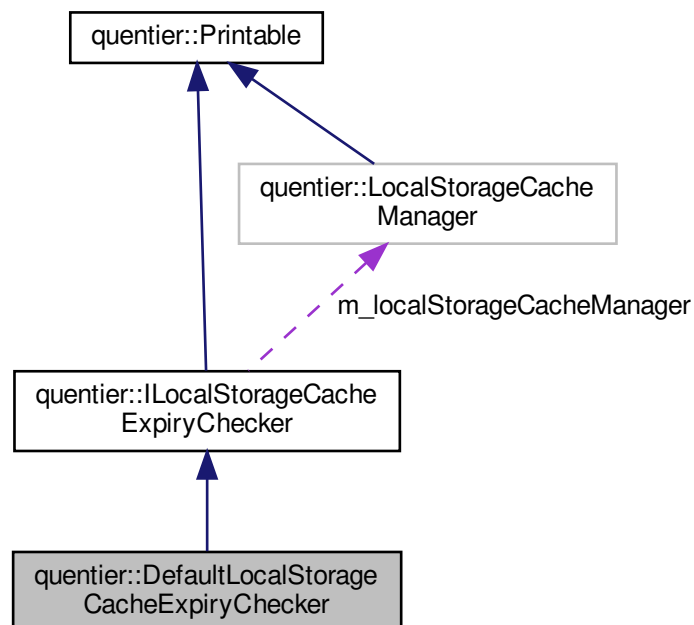
## 5.13 quantier::DefaultLocalStorageCacheExpiryChecker Class Reference

```
#include <DefaultLocalStorageCacheExpiryChecker.h>
```

Inheritance diagram for quantier::DefaultLocalStorageCacheExpiryChecker:



Collaboration diagram for quantier::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

### 5.13.1 Detailed Description

brief The [DefaultLocalStorageCacheExpiryChecker](#) class is the implementation of [ILocalStorageCacheExpiryChecker](#) interface used by [LocalStorageCacheManager](#) by default, if no another implementation of [ILocalStorageCacheExpiryChecker](#) is set to be used by [LocalStorageCacheManager](#)

### 5.13.2 Member Function Documentation

#### 5.13.2.1 [checkLinkedNotebooks\(\)](#)

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

##### Returns

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

Implements [quantier::ILocalStorageCacheExpiryChecker](#).

#### 5.13.2.2 [checkNotebooks\(\)](#)

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

##### Returns

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

Implements [quantier::ILocalStorageCacheExpiryChecker](#).

### 5.13.2.3 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](#).

### 5.13.2.4 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](#).

### 5.13.2.5 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](#).

### 5.13.2.6 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](#).

## 5.13.2.7 clone()

```
virtual DefaultLocalStorageCacheExpiryChecker * quantier::DefaultLocalStorageCacheExpiryChecker::clone ( ) const [override], [virtual]
```

## Returns

A pointer to the newly allocated copy of the current [DefaultLocalStorageCacheExpiryChecker](#)

Implements [quantier::ILocalStorageCacheExpiryChecker](#).

## 5.13.2.8 print()

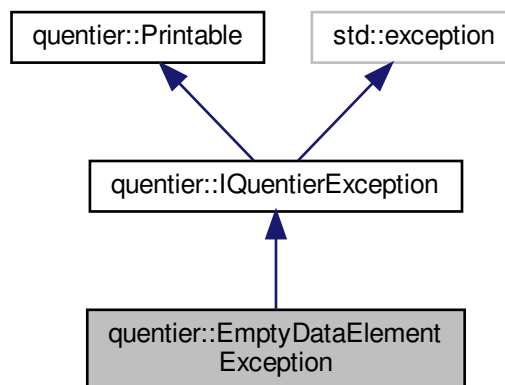
```
virtual QTextStream & quantier::DefaultLocalStorageCacheExpiryChecker::print (
    QTextStream & strm ) const [override], [virtual]
```

Print the internal information about the current [DefaultLocalStorageCacheExpiryChecker](#) instance to the text stream.

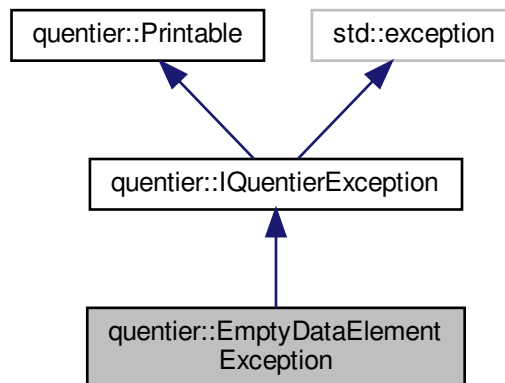
Implements [quantier::ILocalStorageCacheExpiryChecker](#).

## 5.14 quantier::EmptyDataElementException Class Reference

Inheritance diagram for quantier::EmptyDataElementException:



Collaboration diagram for `quentier::EmptyDataElementException`:



## Public Member Functions

- `EmptyDataElementException` (const [ErrorString](#) &message)

## Protected Member Functions

- virtual const `QString` [exceptionDisplayName](#) () const override

### 5.14.1 Member Function Documentation

#### 5.14.1.1 `exceptionDisplayName()`

```
virtual const QString quentier::EmptyDataElementException::exceptionDisplayName ( ) const  
[override], [protected], [virtual]
```

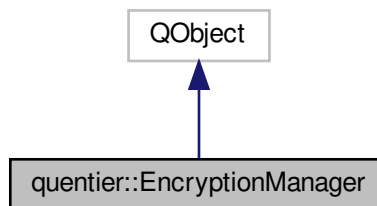
Implements [quentier::IQuentierException](#).

## 5.15 quantier::EncryptionManager Class Reference

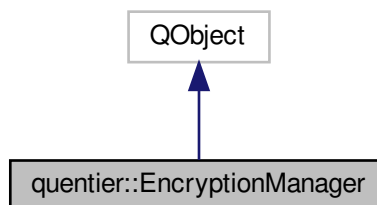
The [EncryptionManager](#) class provides both synchronous methods to encrypt or decrypt given text with password, cipher and key length and their signal-slot based potentially asynchronous counterparts.

```
#include <EncryptionManager.h>
```

Inheritance diagram for quantier::EncryptionManager:



Collaboration diagram for quantier::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)

### 5.15.1 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.

## 5.16 quotienter::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 class [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](#) &error↵Message)
- 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 &ci-pher, 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)

### 5.16.1 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.

### 5.16.2 Member Function Documentation

#### 5.16.2.1 cleanupExternalHtml()

```
bool quantier::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

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

**Returns**

true in case of successful conversion, false otherwise

**5.16.2.2 exportNotesToEnex()**

```
bool quantier::ENMLConverter::exportNotesToEnex (
    const QVector< Note > & notes,
    const QHash< QString, QString > & tagNameByTagLocalUids,
    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**

<i>notes</i>	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
<i>tagNamesByTagLocalUids</i>	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
<i>exportTagsOption</i>	Whether the export to ENEX should include the names of notes' tags
<i>enex</i>	The output of the method
<i>errorDescription</i>	The textual description of the error, if any
<i>version</i>	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

**5.16.2.3 htmlToQTextDocument()**

```
bool quantier::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

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

## Returns

true in case of successful conversion, false otherwise

## 5.16.2.4 importEnex()

```
bool quantier::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

<i>enex</i>	The input ENEX file contents
<i>notes</i>	Notes read from the ENEX
<i>tagNamesByNoteLocalUid</i>	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
<i>errorDescription</i>	The textual description of the error if the ENEX file could not be read and converted into a set of notes and tag names for them

## Returns

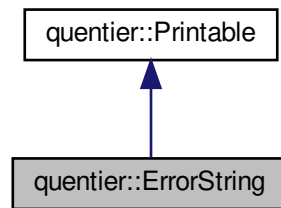
True if the ENEX file was read and converted into a set of notes and tag names successfully, false otherwise

## 5.17 quantier::ErrorString Class Reference

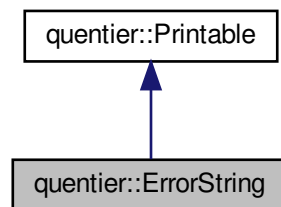
The [ErrorString](#) class encapsulates two (or more) strings which are meant to contain translatable (base) and non-translatable (details) parts of the error description.

```
#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

### 5.17.1 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 return 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

### 5.17.2 Member Function Documentation

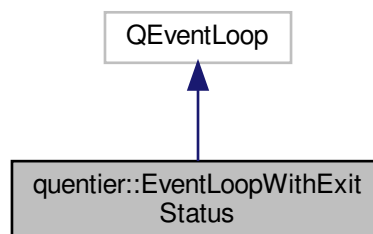
#### 5.17.2.1 print()

```
virtual QTextStream & quantier::ErrorString::print (  
    QTextStream & strm ) const    [override], [virtual]
```

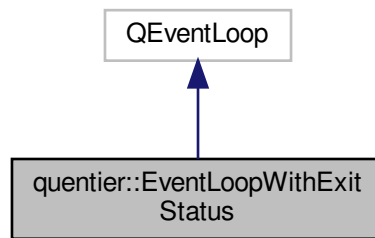
Implements [quantier::Printable](#).

## 5.18 quantier::EventLoopWithExitStatus Class Reference

Inheritance diagram for quantier::EventLoopWithExitStatus:



Collaboration diagram for `quentier::EventLoopWithExitStatus`:



## Public Types

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

## Public Slots

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

## Public Member Functions

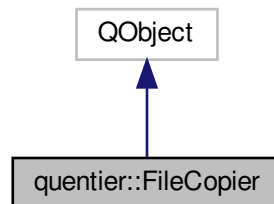
- **EventLoopWithExitStatus** (QObject \*parent=nullptr)
- ExitStatus **exitStatus** () const
- const [ErrorString](#) & **errorDescription** () const

## Friends

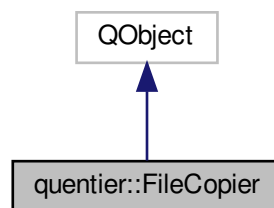
- QDebug & **operator**<< (QDebug &dbg, const ExitStatus status)
- QTextStream & **operator**<< (QTextStream &strm, const ExitStatus status)

## 5.19 quantier::FileCopier Class Reference

Inheritance diagram for quantier::FileCopier:



Collaboration diagram for quantier::FileCopier:



### Public Types

- enum class **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

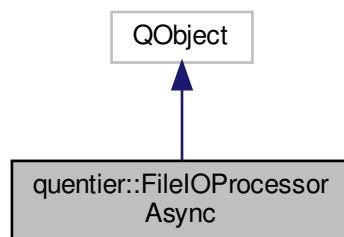
- QDebug & **operator**<< (QDebug &dbg, const State state)
- QTextStream & **operator**<< (QTextStream &strm, const State state)

## 5.20 quantier::FileIOProcessorAsync Class Reference

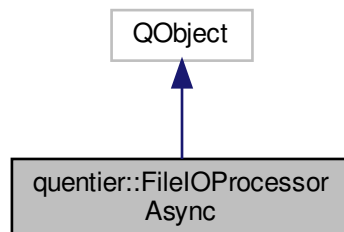
The [FileIOProcessorAsync](#) class is a wrapper under simple file IO operations, it is meant to be used for simple asynchronous IO.

```
#include <FileIOProcessorAsync.h>
```

Inheritance diagram for quantier::FileIOProcessorAsync:



Collaboration diagram for quantier::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.*

### 5.20.1 Detailed Description

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

### 5.20.2 Member Function Documentation

#### 5.20.2.1 onReadFileRequest

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

onReadFileRequest slot processes file read requests with given request ids

#### Parameters

<i>absoluteFilePath</i>	Absolute file path to be read
<i>requestId</i>	Unique identifier of the file read request

### 5.20.2.2 onWriteFileRequest

```
void quantier::FileIOProcessorAsync::onWriteFileRequest (
    QString absoluteFilePath,
    QByteArray data,
    QUuid requestId,
    bool append ) [slot]
```

onWriteFileRequest slot processes file write requests with given request ids

#### Parameters

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

### 5.20.2.3 readFileRequestProcessed

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

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

#### Parameters

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

### 5.20.2.4 setIdleTimePeriod()

```
void quantier::FileIOProcessorAsync::setIdleTimePeriod (
    quint32 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

<i>seconds</i>	Number of seconds for idle time period
----------------	--

### 5.20.2.5 writeFileRequestProcessed

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

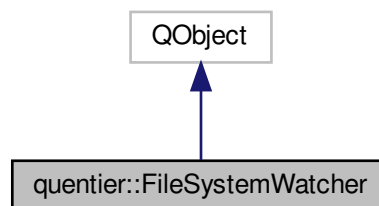
writeFileRequestProcessed signal is emitted when the file write request with given id is finished

## Parameters

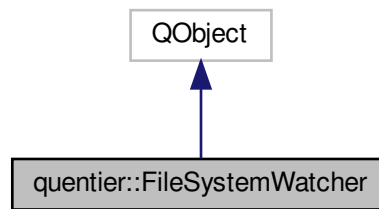
<i>success</i>	True if write operation was successful, false otherwise
<i>errorDescription</i>	Textual description of the error
<i>requestId</i>	Unique identifier of the file write request

## 5.21 quantier::FileSystemWatcher Class Reference

Inheritance diagram for quantier::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)

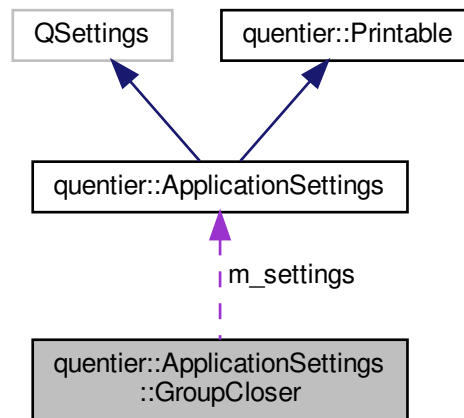
## Public Member Functions

- **FileSystemWatcher** (const int removalTimeoutMsec=FILE\_SYSTEM\_WATCHER\_DEFAULT\_REMOVAL\_TIMEOUT\_MSEC, QObject \*parent=nullptr)
- **FileSystemWatcher** (const QStringList &paths, const int removalTimeoutMsec=FILE\_SYSTEM\_WATCHER\_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)

## 5.22 quentier::ApplicationSettings::GroupCloser Struct Reference

```
#include <ApplicationSettings.h>
```

Collaboration diagram for quantier::ApplicationSettings::GroupCloser:



## Public Member Functions

- **GroupCloser** ([ApplicationSettings](#) &settings)

## Public Attributes

- [ApplicationSettings](#) & **m\_settings**

### 5.22.1 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

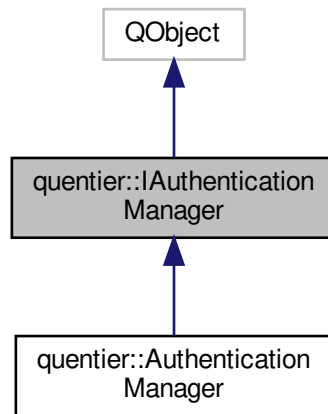
## 5.23 quantier::HTMLCleaner Class Reference

## Public Member Functions

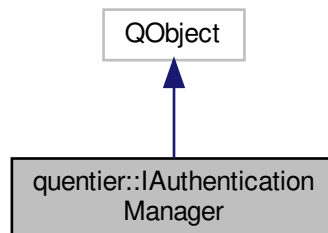
- bool **htmlToXml** (const QString &html, QString &output, QString &errorDescription)
- bool **htmlToXhtml** (const QString &html, QString &output, QString &errorDescription)
- bool **cleanupHtml** (QString &html, QString &errorDescription)

## 5.24 quantier::IAuthenticationManager Class Reference

Inheritance diagram for quantier::IAuthenticationManager:



Collaboration diagram for quantier::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 webApiUrlPrefix, QList< QNetworkCookie > userStoreCookies, [ErrorString](#) errorDescription)

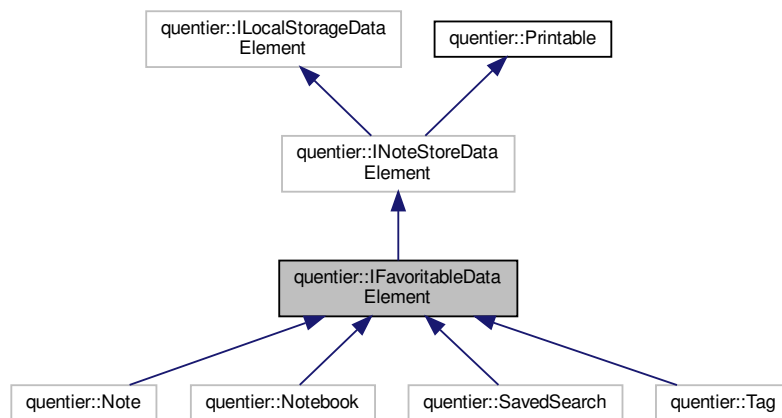
## Protected Member Functions

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

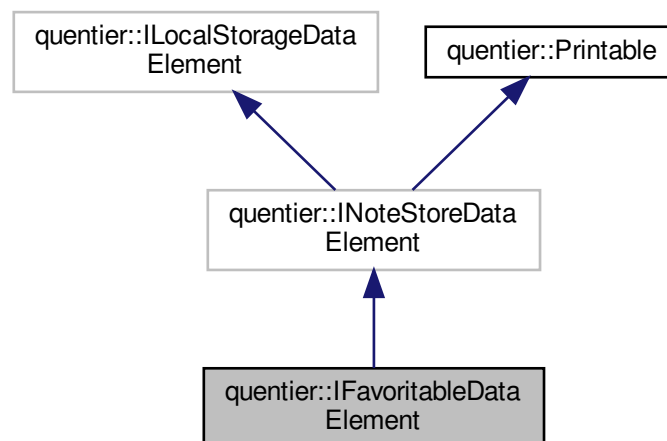
## 5.25 quantier::IFavoritableDataElement Class Reference

```
#include <IFavoritableDataElement.h>
```

Inheritance diagram for quantier::IFavoritableDataElement:



Collaboration diagram for quantier::IFavoritableDataElement:



## Public Member Functions

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

## Additional Inherited Members

### 5.25.1 Detailed Description

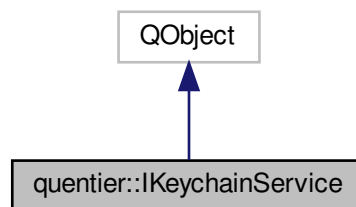
Class adding one more attribute to each note store data element subclassing it: "favorited" flag. Favorited data elements are expected to be somehow arranged for quick access in the client application's UI.

## 5.26 `quentier::IKeychainService` Class Reference

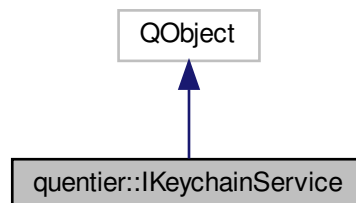
The [IKeychainService](#) interface provides methods intended to start potentially asynchronous interaction with the keychain and signals intended to notify listeners about the completion of asynchronous interactions.

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

Inheritance diagram for `quentier::IKeychainService`:



Collaboration diagram for `quentier::IKeychainService`:





## Public Types

- enum class [ErrorCode](#) {  
[NoError](#) , [EntryNotFound](#) , [CouldNotDeleteEntry](#) , [AccessDeniedByUser](#) ,  
[AccessDenied](#) , [NoBackendAvailable](#) , [NotImplemented](#) , [OtherError](#) }

## Signals

- void [writePasswordJobFinished](#) (QUuid requestId, [ErrorCode](#) errorCode, [ErrorString](#) errorDescription)
- void [readPasswordJobFinished](#) (QUuid requestId, [ErrorCode](#) errorCode, [ErrorString](#) errorDescription, QString 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)

### 5.26.1 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.

### 5.26.2 Member Enumeration Documentation

#### 5.26.2.1 ErrorCode

```
enum class quantier::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

## 5.26.3 Member Function Documentation

### 5.26.3.1 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

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

### 5.26.3.2 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

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

### 5.26.3.3 startDeletePasswordJob()

```
virtual QUuid quotientier::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

<i>service</i>	Name of service within the keychain
<i>key</i>	Key under which the password is stored

#### Returns

Unique identifier assigned to this delete password request

### 5.26.3.4 startReadPasswordJob()

```
virtual QUuid quotientier::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

<i>service</i>	Name of service within the keychain
<i>key</i>	Key under which the password is stored

#### Returns

Unique identifier assigned to this read password request

### 5.26.3.5 startWritePasswordJob()

```
virtual QUuid quotientier::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**

<i>service</i>	Name of service within the keychain
<i>key</i>	Key to store the password under
<i>password</i>	Password to store in the keychain

**Returns**

Unique identifier assigned to this write password request

**5.26.3.6 writePasswordJobFinished**

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

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

**Parameters**

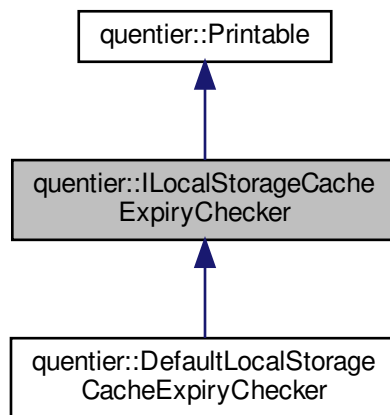
<i>requestId</i>	Request id returned from startWritePasswordJob method
<i>errorCode</i>	Error code determining whether the operation was successful or some error has occurred
<i>errorDescription</i>	Textual description of error in case of unsuccessful execution

**5.27 quantier::ILocalStorageCacheExpiryChecker Class Reference**

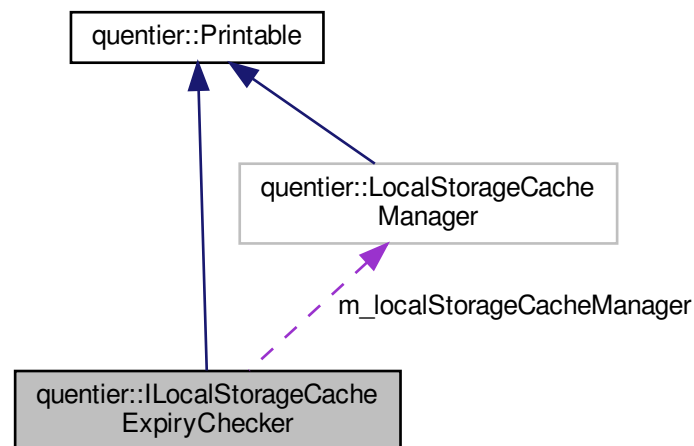
The [ILocalStorageCacheExpiryChecker](#) class represents the interface for cache expiry checker used by [LocalStorageCacheManager](#) to see whether particular caches (of notes, notebooks, tags, linked notebooks and/or saved searches) need to be shrunk.

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

Inheritance diagram for quantier::ILocalStorageCacheExpiryChecker:



Collaboration diagram for quantier::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](#)

### 5.27.1 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.

### 5.27.2 Member Function Documentation

#### 5.27.2.1 [checkLinkedNotebooks\(\)](#)

```
virtual bool quantier::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 [quantier::DefaultLocalStorageCacheExpiryChecker](#).

#### 5.27.2.2 [checkNotebooks\(\)](#)

```
virtual bool quantier::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 [quantier::DefaultLocalStorageCacheExpiryChecker](#).

### 5.27.2.3 `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](#).

### 5.27.2.4 `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](#).

### 5.27.2.5 `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](#).

### 5.27.2.6 `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](#).

### 5.27.2.7 clone()

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

#### Returns

A pointer to the newly allocated copy of a particular [ILocalStorageCacheExpiryChecker](#) implementation

Implemented in [quantier::DefaultLocalStorageCacheExpiryChecker](#).

### 5.27.2.8 print()

```
virtual QTextStream & quantier::ILocalStorageCacheExpiryChecker::print (
    QTextStream & strm ) const [pure virtual]
```

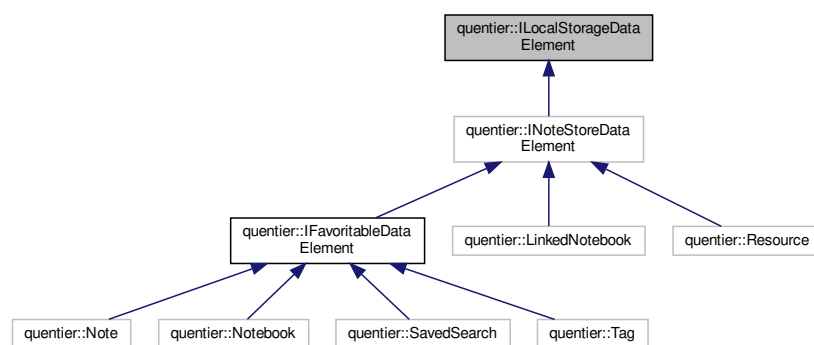
Print the internal information about [ILocalStorageCacheExpiryChecker](#) implementation instance to the text stream.

Implements [quantier::Printable](#).

Implemented in [quantier::DefaultLocalStorageCacheExpiryChecker](#).

## 5.28 quantier::ILocalStorageDataElement Class Reference

Inheritance diagram for [quantier::ILocalStorageDataElement](#):



### Public Member Functions

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

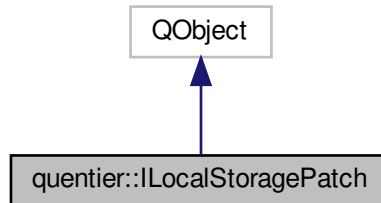


## 5.29 quantier::ILocalStoragePatch Class Reference

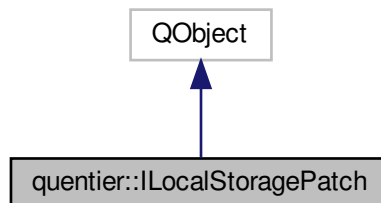
The [ILocalStoragePatch](#) class represents the interface for patches of local storage. Each such patch somehow changes the layout of local storage persistence so that only compliant & corresponding versions of libquantier can be used to work with it.

```
#include <ILocalStoragePatch.h>
```

Inheritance diagram for quantier::ILocalStoragePatch:



Collaboration diagram for quantier::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`

### 5.29.1 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.

### 5.29.2 Member Function Documentation

#### 5.29.2.1 `apply()`

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

Apply the patch to local storage

##### Parameters

<i>errorDescription</i>	The textual description of the error in case of patch application failure
-------------------------	---

##### Returns

True in case of successful patch application, false otherwise

#### 5.29.2.2 `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

<i>errorDescription</i>	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

**5.29.2.3 backupProgress**

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

Local storage backup preparation progress

**Parameters**

<i>progress</i>	Backup preparation progress value, from 0 to 1
-----------------	--

**5.29.2.4 fromVersion()**

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

**Returns**

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

**5.29.2.5 patchLongDescription()**

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

**Returns**

Long i.e. detailed description of the patch

**5.29.2.6 patchShortDescription()**

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

**Returns**

Short description of the patch

**5.29.2.7 progress**

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

Patch application progress signal

## Parameters

<i>progress</i>	Patch application progress value, from 0 to 1
-----------------	---

**5.29.2.8 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

<i>errorDescription</i>	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

**5.29.2.9 restoreBackupProgress**

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

Local storage restoration from backup progress

## Parameters

<i>progress</i>	Local storage restoration from backup progress value, from 0 to 1
-----------------	---

**5.29.2.10 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

<i>errorDescription</i>	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

**5.29.2.11 toVersion()**

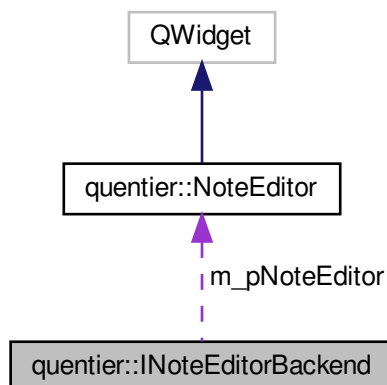
```
virtual int quantier::ILocalStoragePatch::toVersion ( ) const [pure virtual]
```

**Returns**

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

## 5.30 quantier::INoteEditorBackend Class Reference

Collaboration diagram for quantier::INoteEditorBackend:

**Public Types**

- enum class **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

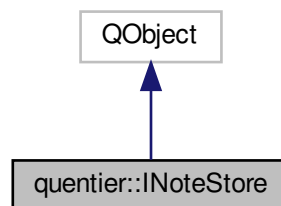
- QUENTIER\_EXPORT QTextStream & **operator**<< (QTextStream &strm, const Rotation rotation)
- QUENTIER\_EXPORT QDebug & **operator**<< (QDebug &dbg, const Rotation rotation)

## 5.31 quentier::INoteStore Class Reference

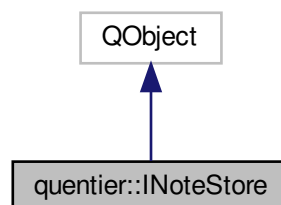
[INoteStore](#) is the interface which provides methods required for the implementation of NoteStore part of Evernote EDAM sync protocol.

```
#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 rateLimitSeconds, [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 &rateLimitSeconds, QString linkedNotebookAuthToken={})=0
- virtual qint32 **updateNotebook** ([Notebook](#) &notebook, [ErrorString](#) &errorDescription, qint32 &rateLimitSeconds, 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 linkedNotebookAuthToken={})=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 &rateLimitSeconds)=0
- virtual qint32 **getSyncChunk** (const qint32 afterUSN, const qint32 maxEntries, const qevercloud::SyncChunkFilter &filter, qevercloud::SyncChunk &syncChunk, [ErrorString](#) &errorDescription, qint32 &rateLimitSeconds)=0
- virtual qint32 **getLinkedNotebookSyncState** (const qevercloud::LinkedNotebook &linkedNotebook, const QString &authToken, qevercloud::SyncState &syncState, [ErrorString](#) &errorDescription, qint32 &rateLimitSeconds)=0
- virtual qint32 **getLinkedNotebookSyncChunk** (const qevercloud::LinkedNotebook &linkedNotebook, const qint32 afterUSN, const qint32 maxEntries, const QString &linkedNotebookAuthToken, const bool fullSyncOnly, qevercloud::SyncChunk &syncChunk, [ErrorString](#) &errorDescription, qint32 &rateLimitSeconds)=0
- virtual qint32 **getNote** (const bool withContent, const bool withResourcesData, const bool withResourcesRecognition, const bool withResourceAlternateData, [Note](#) &note, [ErrorString](#) &errorDescription, qint32 &rateLimitSeconds)=0
- virtual bool **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)=0
- virtual qint32 **getResource** (const bool withDataBody, const bool withRecognitionDataBody, const bool withAlternateDataBody, 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)

### 5.31.1 Detailed Description

[INoteStore](#) is the interface which provides methods required for the implementation of NoteStore part of Evernote EDAM sync protocol.

### 5.31.2 Member Function Documentation

#### 5.31.2.1 authenticateToSharedNotebook()

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

Authenticate to shared notebook

##### Parameters

<i>shareKey</i>	The shared notebook global identifier
<i>authResult</i>	Output parameter, the result of authentication
<i>errorDescription</i>	The textual description of the error if authentication to shared notebook could not be performed
<i>rateLimitSeconds</i>	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

#### 5.31.2.2 createNote()

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

Create note

## Parameters

<i>note</i>	<a href="#">Note</a> to be created
<i>errorDescription</i>	The textual description of the error in case note could not be created
<i>rateLimitSeconds</i>	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
<i>linkedNotebookAuthToken</i>	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

## 5.31.2.3 createNotebook()

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

## Create notebook

## Parameters

<i>notebook</i>	<a href="#">Notebook</a> to be created, must have name set and either "active" or "default notebook" fields may be set
<i>errorDescription</i>	The textual description of the error in case notebook could not be created
<i>rateLimitSeconds</i>	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
<i>linkedNotebookAuthToken</i>	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::EDAMErrorCode enumeration instead

## 5.31.2.4 createSavedSearch()

```
virtual qint32 quantier::INoteStore::createSavedSearch (
    SavedSearch & savedSearch,
```

```

    ErrorString & errorDescription,
    qint32 & rateLimitSeconds ) [pure virtual]

```

Create saved search

#### Parameters

<i>savedSearch</i>	Saved search to be created, must have name and query set, can also have search scope set
<i>errorDescription</i>	The textual description of the error in case saved search could not be created
<i>rateLimitSeconds</i>	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 <code>qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED</code>

#### Returns

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

### 5.31.2.5 createTag()

```

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

```

Create tag

#### Parameters

<i>note</i>	<a href="#">Tag</a> to be created, must have name set, can also have parent guid set
<i>errorDescription</i>	The textual description of the error in case tag could not be created
<i>rateLimitSeconds</i>	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 <code>qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED</code>
<i>linkedNotebookAuthToken</i>	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

## 5.31.2.6 getLinkedNotebookSyncChunk()

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

Get linked notebook sync chunk

## Parameters

<i>linkedNotebook</i>	The linked notebook for which the sync chunk is being retrieved, must contain identifying information and permissions to access the notebook in question
<i>afterUSN</i>	The USN after which the sync chunks are being requested
<i>maxEntries</i>	Max number of items within the sync chunk to be returned
<i>linkedNotebookAuthToken</i>	The authentication token to use for the data from the linked notebook
<i>fullSyncOnly</i>	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
<i>syncChunk</i>	Output parameter, the sync chunk
<i>errorDescription</i>	The textual description of the error in case linked notebook sync chunk could not be retrieved
<i>rateLimitSeconds</i>	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 <code>qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED</code>

## Returns

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

## 5.31.2.7 getLinkedNotebookSyncState()

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

Get linked notebook sync state

## Parameters

<i>linkedNotebook</i>	The linked notebook for which the sync state is being retrieved, must contain identifying information and permissions to access the notebook in question
<i>authToken</i>	The authentication token to use for the data from the linked notebook
<i>syncState</i>	Output parameter, the sync state
<i>errorDescription</i>	The textual description of the error in case linked notebook sync state could not be retrieved
<i>rateLimitSeconds</i>	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 <code>qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED</code>

## Returns

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

5.31.2.8 `getNote()`

```
virtual qint32 qentier::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

<i>withContent</i>	If true, the returned note would include the content
<i>withResourcesData</i>	If true, any resources the note might have would include their full data
<i>withResourcesRecognition</i>	If true, any resources the note might have and which have Evernote supplied recognition would include their full recognition data
<i>withResourceAlternateData</i>	If true, any resources the note might have would include their full alternate data
<i>note</i>	Input and output parameter, the retrieved note, must have guid set
<i>errorDescription</i>	The textual description of the error in case the note could not be retrieved
<i>rateLimitSeconds</i>	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 <code>qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED</code>

## Returns

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

### 5.31.2.9 getNoteAsync()

```
virtual bool quantier::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,
    QString & errorDescription ) [pure virtual]
```

Get note asynchronously

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

#### Parameters

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

### 5.31.2.10 getResource()

```
virtual qint32 quantier::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

<i>withDataBody</i>	If true, the returned resource would include its data body
<i>withRecognitionDataBody</i>	If true, the returned resource would include its recognition data body
<i>withAlternateDataBody</i>	If true, the returned resource would include its alternate data body
<i>withAttributes</i>	If true, the returned resource would include its attributes
<i>authToken</i>	Authentication token to use for resource retrieval
<i>resource</i>	Input and output parameter, the retrieved resource, must have guid set
<i>errorDescription</i>	The textual description of the error in case the resource could not be retrieved
<i>rateLimitSeconds</i>	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::EDAMError↵ Code enumeration instead

#### 5.31.2.11 getResourceAsync()

```
virtual bool qentier::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 invocation would be returned via the emission of get↵ ResourceAsyncFinished signal.

#### Parameters

<i>withDataBody</i>	If true, the returned resource would include its data body
<i>withRecognitionDataBody</i>	If true, the returned resource would include its recognition data body
<i>withAlternateDataBody</i>	If true, the returned resource would include its alternate data body
<i>withAttributes</i>	If true, the returned resource would include its attributes
<i>resourceGuid</i>	The guid of the resource to be retrieved
<i>authToken</i>	Authentication token to use for resource retrieval
<i>errorDescription</i>	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

## 5.31.2.12 getSyncChunk()

```
virtual qint32 quotientier::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

<i>afterUSN</i>	The USN after which the sync chunks are being requested
<i>maxEntries</i>	Max number of items within the sync chunk to be returned
<i>filter</i>	Filter for items to be returned within the sync chunks
<i>syncChunk</i>	Output parameter, the sync chunk
<i>errorDescription</i>	The textual description of the error in case sync chunk could not be retrieved
<i>rateLimitSeconds</i>	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::↵  
EDAMErrorCode enumeration instead

## 5.31.2.13 getSyncState()

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

Get sync state

## Parameters

<i>syncState</i>	Output parameter, the sync state
<i>errorDescription</i>	The textual description of the error in case sync state could not be retrieved
<i>rateLimitSeconds</i>	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::EDAMErrorCode` enumeration instead

**5.31.2.14 noteStoreUrl()**

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

Provide note store URL used by this [INoteStore](#) instance

**5.31.2.15 setAuthData()**

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

Set authentication data to be used by this [INoteStore](#) instance

**5.31.2.16 setNoteStoreUrl()**

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

Set note store URL to be used by this [INoteStore](#) instance

**5.31.2.17 stop()**

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

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

**5.31.2.18 updateNote()**

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

Update note

**Parameters**

<i>note</i>	<a href="#">Note</a> to be updated, must have guid set
<i>errorDescription</i>	The textual description of the error in case note could not be updated
<i>rateLimitSeconds</i>	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 <code>qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED</code>
<i>linkedNotebookAuthToken</i>	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

## 5.31.2.19 updateNotebook()

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

Update notebook

## Parameters

<i>notebook</i>	<a href="#">Notebook</a> to be updated, must have guid set
<i>errorDescription</i>	The textual description of the error in case notebook could not be updated
<i>rateLimitSeconds</i>	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 <code>qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED</code>
<i>linkedNotebookAuthToken</i>	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

## 5.31.2.20 updateSavedSearch()

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

Update saved search

## Parameters

<i>savedSearch</i>	Saved search to be updated, must have guid set
<i>errorDescription</i>	The textual description of the error in case saved search could not be updated
<i>rateLimitSeconds</i>	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 <code>qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED</code>

**Returns**

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

**5.31.2.21 updateTag()**

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

**Update tag****Parameters**

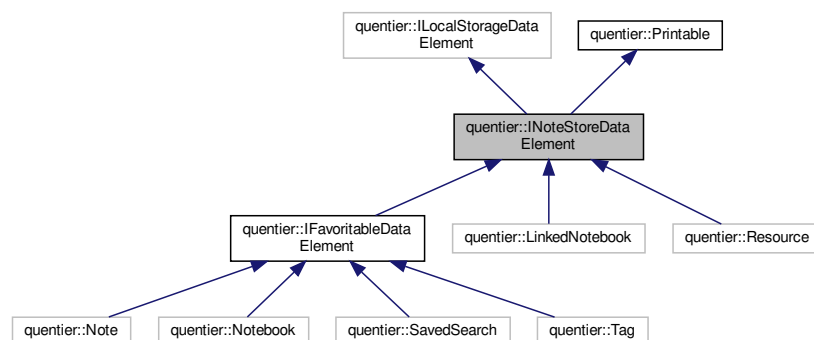
<i>tag</i>	<a href="#">Tag</a> to be updated, must have guid set
<i>errorDescription</i>	The textual description of the error in case tag could not be updated
<i>rateLimitSeconds</i>	Output parameter, the number of seconds the client needs to wait before attempting to call this method or any other method calling Evernote API again; only meaningful if returned value matches <code>qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED</code>
<i>linkedNotebookAuthToken</i>	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**

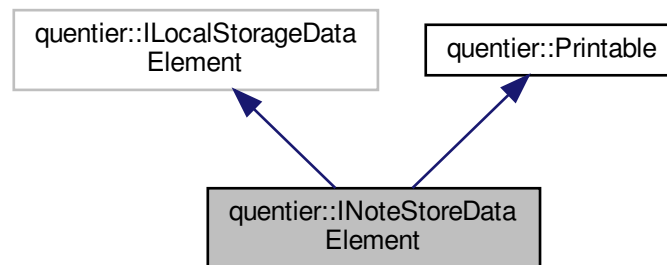
Error code, 0 in case of successful tag update, other values corresponding to `qevercloud::EDAMErrorCode` enumeration instead

**5.32 quantier::INoteStoreDataElement Class Reference**

Inheritance diagram for `quantier::INoteStoreDataElement`:



Collaboration diagram for quantier::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

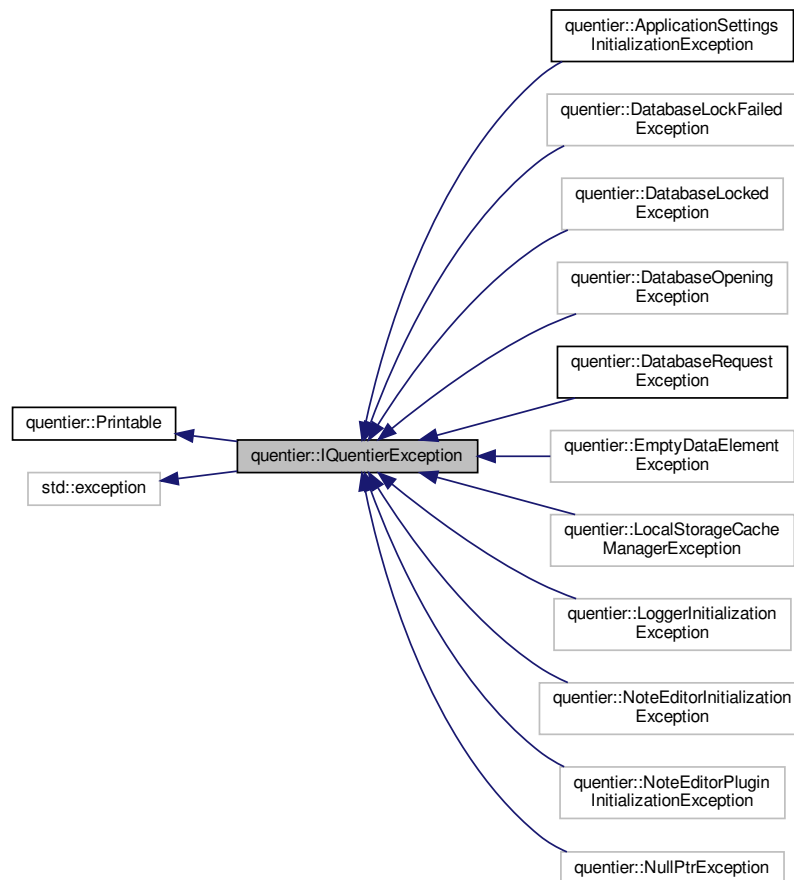
## Additional Inherited Members

## 5.33 quantier::IQuantierException Class Reference

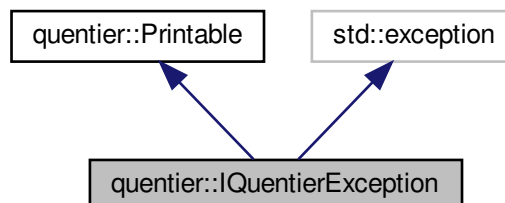
The [IQuantierException](#) class represents the interface for exceptions specific to libquantier and applications based on it.

```
#include <IQuantierException.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

### 5.33.1 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.

### 5.33.2 Member Function Documentation

#### 5.33.2.1 `print()`

```
virtual QTextStream & quentier::IQuentierException::print (
    QTextStream & strm ) const [override], [virtual]
```

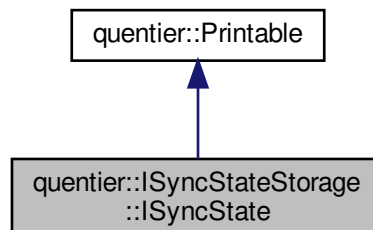
Implements `quentier::Printable`.

## 5.34 `quentier::ISyncStateStorage::ISyncState` Class Reference

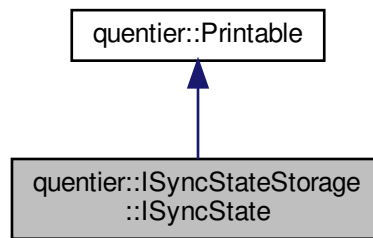
The `ISyncState` interface provides accessory methods to determine the sync state for the account.

```
#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

### 5.34.1 Detailed Description

The [ISyncState](#) interface provides accessory methods to determine the sync state for the account.

### 5.34.2 Member Function Documentation

#### 5.34.2.1 print()

```
virtual QTextStream & quentier::ISyncStateStorage::ISyncState::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [quentier::Printable](#).

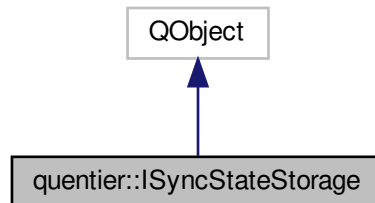


## 5.35 quantier::ISyncStateStorage Class Reference

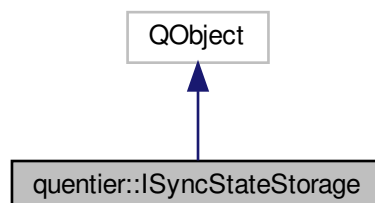
The [ISyncStateStorage](#) interface represents the interface of a class which stores sync state for given accounts persistently and provides access to previously stores sync states.

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

Inheritance diagram for quantier::ISyncStateStorage:



Collaboration diagram for quantier::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

### 5.35.1 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.

### 5.35.2 Member Function Documentation

#### 5.35.2.1 notifySyncStateUpdated

```
void quantier::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

## 5.36 quantier::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

### 5.36.1 Detailed Description

[IUserStore](#) is the interface which provides methods required for the implementation of UserStore part of Evernote EDAM sync protocol.

## 5.36.2 Member Function Documentation

### 5.36.2.1 checkVersion()

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

Check the version of EDAM protocol

#### Parameters

<i>clientName</i>	Application name + application version + platform name string
<i>edamVersionMajor</i>	The major version of EDAM protocol the application wants to use to connect to Evernote
<i>edamVersionMinor</i>	The minor version of EDAM protocol the application wants to use to connect to Evernote
<i>errorDescription</i>	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

### 5.36.2.2 getAccountLimits()

```
virtual qint32 quantier::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

<i>serviceLevel</i>	The level of Evernote service for which account limits are requested
<i>limits</i>	Output account limits
<i>errorDescription</i>	The textual description of the error if account limits could not be retrieved
<i>rateLimitSeconds</i>	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 `qevercloud::EDAMErrorCode::type` enumeration instead

**5.36.2.3 getUser()**

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

Retrieve full information about user (account)

**Parameters**

<i>user</i>	Input and output parameter; on input needs to have user id set
<i>errorDescription</i>	The textual description of the error if full user information could not be retrieved
<i>rateLimitSeconds</i>	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 <code>qevercloud::EDAMErrorCode::RATE_LIMIT_REACHED</code>

**Returns**

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

**5.36.2.4 setAuthData()**

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

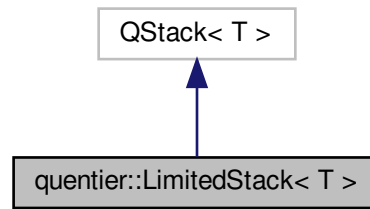
Set authentication data to be used by this [IUserStore](#) instance

**5.37 qentier::LimitedStack< T > Class Template Reference**

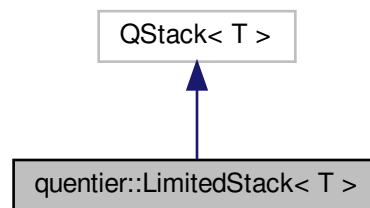
The [LimitedStack](#) template class implements a stack which may have a limitation for its size; when the size becomes too much according to the limit, the bottom element of the stack gets erased from it. Only limits greater than zero are considered.

```
#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`)

### 5.37.1 Detailed Description

```

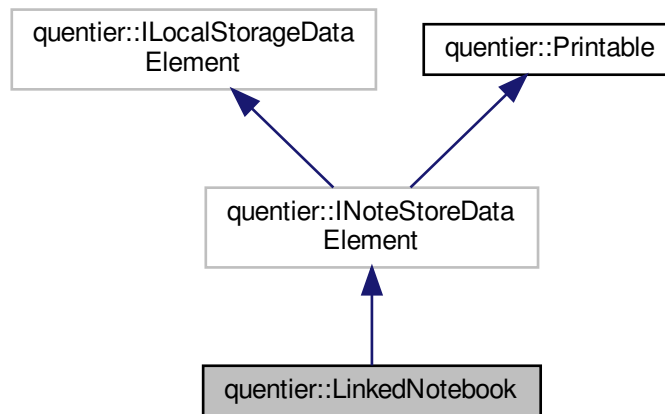
template<class T>
class quentier::LimitedStack< T >

```

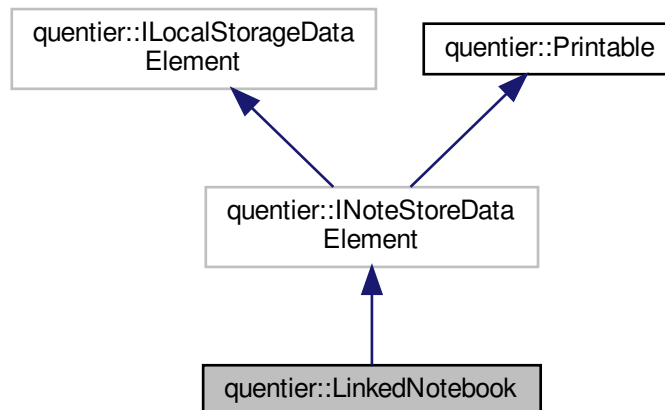
The `LimitedStack` template class implements a stack which may have a limitation for its size; when the size becomes too much according to the limit, the bottom element of the stack gets erased from it. Only limits greater than zero are considered.

## 5.38 quantier::LinkedNotebook Class Reference

Inheritance diagram for quantier::LinkedNotebook:



Collaboration diagram for quantier::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** (`QString` &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

### 5.38.1 Member Function Documentation

#### 5.38.1.1 checkParameters()

```
virtual bool quentier::LinkedNotebook::checkParameters (
    ErrorString & errorDescription ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.38.1.2 clear()

```
virtual void quentier::LinkedNotebook::clear ( ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.38.1.3 guid()

```
virtual const QString & quentier::LinkedNotebook::guid ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.38.1.4 hasGuid()

```
virtual bool quentier::LinkedNotebook::hasGuid ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.38.1.5 hasUpdateSequenceNumber()

```
virtual bool quentier::LinkedNotebook::hasUpdateSequenceNumber ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.38.1.6 print()

```
virtual QTextStream & quentier::LinkedNotebook::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [quentier::Printable](#).



#### 5.38.1.7 setGuid()

```
virtual void quantier::LinkedNotebook::setGuid (
    const QString & guid ) [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.38.1.8 setUpdateSequenceNumber()

```
virtual void quantier::LinkedNotebook::setUpdateSequenceNumber (
    const quint32 usn ) [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

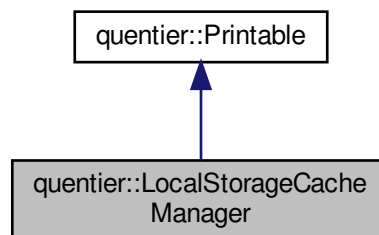
#### 5.38.1.9 updateSequenceNumber()

```
virtual quint32 quantier::LinkedNotebook::updateSequenceNumber ( ) const [override], [virtual]
```

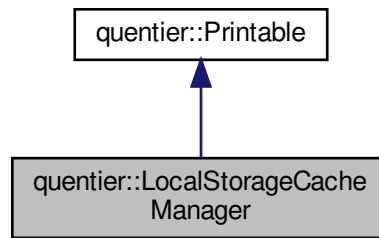
Implements [quantier::INoteStoreDataElement](#).

## 5.39 quantier::LocalStorageCacheManager Class Reference

Inheritance diagram for quantier::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

### 5.39.1 Member Function Documentation

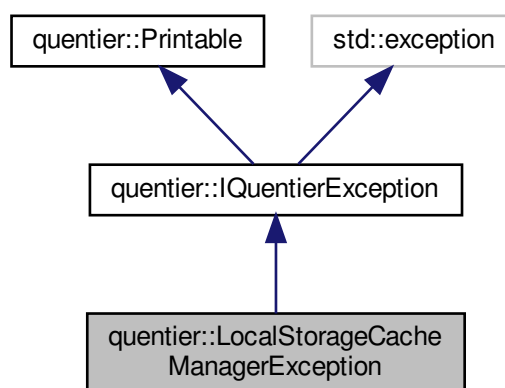
#### 5.39.1.1 `print()`

```
virtual QTextStream & quentier::LocalStorageCacheManager::print (
    QTextStream & strm ) const [override], [virtual]
```

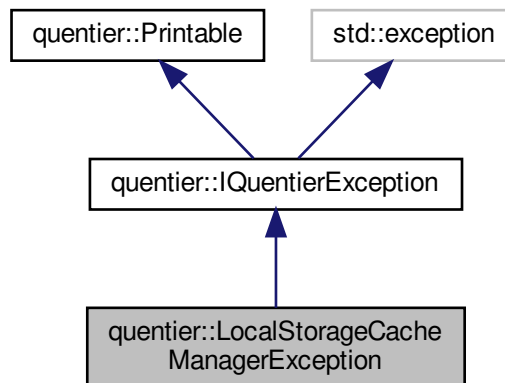
Implements [quentier::Printable](#).

## 5.40 `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

### 5.40.1 Member Function Documentation

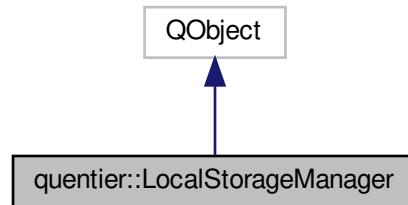
#### 5.40.1.1 `exceptionDisplayName()`

```
virtual const QString quentier::LocalStorageCacheManagerException::exceptionDisplayName ( )  
const [override], [protected], [virtual]
```

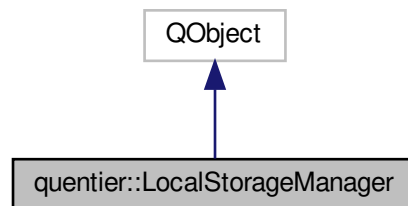
Implements [quentier::IQuentierException](#).

## 5.41 quantier::LocalStorageManager Class Reference

Inheritance diagram for quantier::LocalStorageManager:



Collaboration diagram for quantier::LocalStorageManager:



### Public Types

- enum class [StartupOption](#) { `ClearDatabase` = 1 , `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 class [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 class [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 class [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 class `ListLinkedNotebooksOrder` { `ByUpdateSequenceNumber` = 0 , `ByShareName` , `ByUsername` , `NoOrder` }

The `ListLinkedNotebooksOrder` enum allows to specify the results ordering for methods listing linked notebooks from local storage.

- enum class `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 class `UpdateNoteOption` { `UpdateResourceMetadata` = 1 , `UpdateResourceBinaryData` = 2 , `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 class `GetNoteOption` { `WithResourceMetadata` = 1 , `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 class `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 class `ListTagsOrder` { `ByUpdateSequenceNumber` , `ByName` , `NoOrder` }

The `ListTagsOrder` enum allows to specify the results ordering for methods listing tags from the local storage database.

- enum class `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 class `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 passed 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)`
- `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 > &noteCountsPerTagLocalUids, 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)`
- `bool updateNote (Note &note, const UpdateNoteOptions options, ErrorString &errorDescription)`
- `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 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



- `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](#) &orderDirection=OrderDirection::Ascending) const  
*listNotesByLocalUids attempts to list notes given their local uids*
- `QList< Note > 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.*
- `QStringList findNoteLocalUidsWithSearchQuery` (const [NoteSearchQuery](#) &noteSearchQuery, [ErrorString](#) &errorDescription) const  
*findNoteLocalUidsWithSearchQuery attempts to find note local uids of notes corresponding to the passed in [NoteSearchQuery](#) 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 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*
- `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](#) orderDirection=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, [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*
- `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 searches 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)`

## 5.41.1 Member Enumeration Documentation

### 5.41.1.1 `GetNoteOption`

```
enum class 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

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

### 5.41.1.2 GetResourceOption

```
enum class 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
----------------	--

### 5.41.1.3 ListObjectsOption

```
enum class 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.

### 5.41.1.4 StartupOption

```
enum class 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, <a href="#">LocalStorageManager</a> would wipe any existing database contents; the net effect would be as if no database existed for the given user before the creation of <a href="#">LocalStorageManager</a> or before the call to its switchUser method
OverrideLock	If OverrideLock flag is active, <a href="#">LocalStorageManager</a> would ignore the existing advisory lock (if any) put on the database file; if this flag is not active, the attempt to create <a href="#">LocalStorageManager</a> (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 <a href="#">DatabaseLockedException</a>

### 5.41.1.5 UpdateNoteOption

```
enum class 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

## 5.41.2 Constructor & Destructor Documentation

### 5.41.2.1 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

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

## 5.41.3 Member Function Documentation

### 5.41.3.1 accountHighUsn()

```
qint32 quantier::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

<i>linkedNotebookGuid</i>	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
<i>errorDescription</i>	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

### 5.41.3.2 addEnResource()

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

addEnResource adds passed in resource to the local storage database.

#### Parameters

<i>resource</i>	<a href="#">Resource</a> 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
<i>errorDescription</i>	Error description if resource could not be added

#### Returns

True if resource was added successfully, false otherwise

### 5.41.3.3 addLinkedNotebook()

```
bool quantier::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

<i>linkedNotebook</i>	<a href="#">LinkedNotebook</a> to be added to the local storage database
<i>errorDescription</i>	Error description if linked notebook could not be added

## Returns

True if linked notebook was added successfully, false otherwise

## 5.41.3.4 addNote()

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

addNote adds passed in [Note](#) to the local storage database.

## Parameters

<i>note</i>	<a href="#">Note</a> 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
<i>errorDescription</i>	Error description if note could not be added

## Returns

True if note was added successfully, false otherwise

## 5.41.3.5 addNotebook()

```
bool quantier::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

<i>notebook</i>	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)
<i>errorDescription</i>	Error description if the notebook could not be added

**Returns**

True if the notebook was added successfully, false otherwise

**5.41.3.6 addSavedSearch()**

```
bool quantier::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**

<i>search</i>	<a href="#">SavedSearch</a> 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
<i>errorDescription</i>	Error description if <a href="#">SavedSearch</a> could not be added

**Returns**

True if [SavedSearch](#) was added successfully, false otherwise

**5.41.3.7 addTag()**

```
bool quantier::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**

<i>tag</i>	<a href="#">Tag</a> 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
<i>errorDescription</i>	Error description if <a href="#">Tag</a> could not be added

**Returns**

True if [Tag](#) was added successfully, false otherwise



### 5.41.3.8 addUser()

```
bool quantier::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

<i>user</i>	The user to be added to the local storage database
<i>errorDescription</i>	Error description if the user could not be added

#### Returns

True if the user was added successfully, false otherwise

### 5.41.3.9 deleteUser()

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

deleteUser marks the user as deleted in local storage

#### Parameters

<i>user</i>	The user to be marked as deleted
<i>errorDescription</i>	Error description if the user could not be marked as deleted

#### Returns

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

### 5.41.3.10 enResourceCount()

```
int quantier::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**

<i>errorDescription</i>	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

**5.41.3.11 expungeEnResource()**

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

expungeResource permanently deletes resource from the local storage database.

**Parameters**

<i>resource</i>	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
<i>errorDescription</i>	Error description if resource could not be expunged

**Returns**

True if resource was expunged successfully, false otherwise

**5.41.3.12 expungeLinkedNotebook()**

```
bool quantier::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 client.

**Parameters**

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

**Returns**

True if linked notebook was expunged successfully, false otherwise

**5.41.3.13 expungeNote()**

```
bool quantier::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**

<i>note</i>	<a href="#">Note</a> to be expunged; may be changed as a result of the call, filled with fields like local uid or notebook guid or local uid
<i>errorDescription</i>	Error description if note could not be expunged

**Returns**

True if note was expunged successfully, false otherwise

**5.41.3.14 expungeNotebook()**

```
bool quantier::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**

<i>notebook</i>	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)
<i>errorDescription</i>	Error description if the notebook could not be expunged

**Returns**

True if the notebook was expunged successfully, false otherwise

**5.41.3.15 expungeNotelessTagsFromLinkedNotebooks()**

```
bool quantier::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**

<i>errorDescription</i>	Error description if tag could not be expunged
-------------------------	--

**Returns**

True if relevant tags were expunged successfully, false otherwise

**5.41.3.16 expungeSavedSearch()**

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

expungeSavedSearch permanently deletes saved search from the local storage database.

**Parameters**

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

**Returns**

True if saved search was expunged successfully, false otherwise

**5.41.3.17 expungeTag()**

```
bool quantier::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

<i>tag</i>	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
<i>expungedChildTagLocalUids</i>	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
<i>errorDescription</i>	Error description if tag could not be expunged

#### Returns

True if tag was expunged successfully, false otherwise

### 5.41.3.18 expungeUser()

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

expungeUser permanently deletes the user from the local storage database

#### Parameters

<i>user</i>	The user to be expunged
<i>errorDescription</i>	Error description if the user could not be expunged

#### Returns

True if the user was expunged successfully, false otherwise

### 5.41.3.19 findDefaultNotebook()

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

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

## Parameters

<i>notebook</i>	The default notebook to be found
<i>errorDescription</i>	Error description if the default notebook could not be found

## Returns

True if the default notebook was found, false otherwise

**5.41.3.20 findDefaultOrLastUsedNotebook()**

```
bool quantier::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

<i>notebook</i>	Either the default or the last used notebook to be found
<i>errorDescription</i>	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

**5.41.3.21 findEnResource()**

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

findEnResource method attempts to find resource in the local storage database

## Parameters

<i>resource</i>	<a href="#">Resource</a> 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
<i>options</i>	Options specifying which optionally includable fields of the resource should actually be included
<i>errorDescription</i>	Error description if resource could not be found

**Returns**

True if resource was found successfully, false otherwise

**5.41.3.22 findLastUsedNotebook()**

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

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

**Parameters**

<i>notebook</i>	The last used notebook to be found
<i>errorDescription</i>	Error description if the last used notebook could not be found

**Returns**

True if the last used notebook was found, false otherwise

**5.41.3.23 findLinkedNotebook()**

```
bool quantier::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**

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

**Returns**

True if linked notebook was found, false otherwise

### 5.41.3.24 findNote()

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

findNote - attempts to find note in the local storage database

#### Parameters

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

#### Returns

true if note was found successfully, false otherwise

### 5.41.3.25 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

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

#### Returns

True if the notebook was found, false otherwise



### 5.41.3.26 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

<i>noteSearchQuery</i>	Filled <a href="#">NoteSearchQuery</a> object used to filter the notes
<i>errorDescription</i>	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

### 5.41.3.27 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

<i>noteSearchQuery</i>	Filled <a href="#">NoteSearchQuery</a> object used to filter the notes
<i>options</i>	Options specifying which optionally includable fields of the note should actually be included
<i>errorDescription</i>	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](#)

### 5.41.3.28 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

<i>search</i>	<a href="#">SavedSearch</a> to be found in the local storage database
<i>errorDescription</i>	Error description if <a href="#">SavedSearch</a> could not be found

## Returns

True if [SavedSearch](#) was found, false otherwise

**5.41.3.29 findTag()**

```
bool quantier::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

<i>tag</i>	<a href="#">Tag</a> to be found in the local storage database; must have either guid, local uid or name set
<i>errorDescription</i>	Error description in case tag could not be found

## Returns

True if tag was found, false otherwise

**5.41.3.30 findUser()**

```
bool quantier::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

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

## Returns

True if the user was found successfully, false otherwise

**5.41.3.31 highestSupportedLocalStorageVersion()**

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

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

## Returns

Highest supported local storage version

**5.41.3.32 isLocalStorageVersionTooHigh()**

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

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

NOTE: it is libquantier client code's responsibility to call this method and/or localStorageRequiresUpgrade method, libquantier 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

<i>errorDescription</i>	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 libquantier 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 libquantier to work with, false otherwise

### 5.41.3.33 linkedNotebookCount()

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

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

#### Parameters

<i>errorDescription</i>	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

### 5.41.3.34 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

<i>errorDescription</i>	Error description if linked notebooks could not be listed, otherwise this parameter is untouched
<i>limit</i>	Limit for the max number of linked notebooks in the result, zero by default which means no limit is set
<i>offset</i>	Number of linked notebooks to skip in the beginning of the result, zero by default
<i>order</i>	Allows to specify particular ordering of linked notebooks in the result, NoOrder by default
<i>orderDirection</i>	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

### 5.41.3.35 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

<i>errorDescription</i>	Error description if all notebooks could not be listed; if no error happens, this parameter is untouched
<i>limit</i>	The limit for the max number of notebooks in the result, zero by default which means no limit is set
<i>offset</i>	The number of notebooks to skip in the beginning of the result, zero by default
<i>order</i>	Allows to specify a particular ordering of notebooks in the result, NoOrder by default
<i>orderDirection</i>	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder
<i>linkedNotebookGuid</i>	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

#### 5.41.3.36 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

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

**5.41.3.37 listAllSharedNotebooks()**

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

listAllSharedNotebooks attempts to list all shared notebooks within the account.

**Parameters**

<i>errorDescription</i>	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

**5.41.3.38 listAllTags()**

```
QList< Tag > quantier::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**

<i>errorDescription</i>	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.
<i>limit</i>	Limit for the max number of tags in the result, zero by default which means no limit is set
<i>offset</i>	Number of tags to skip in the beginning of the result, zero by default
<i>order</i>	Allows to specify particular ordering of tags in the result, NoOrder by default
<i>orderDirection</i>	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder
<i>linkedNotebookGuid</i>	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

**5.41.3.39 `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**

<i>note</i>	<code>Note</code> 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.
<i>errorDescription</i>	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.
<i>flag</i>	Input parameter used to set the filter for the desired tags to be listed
<i>limit</i>	Limit for the max number of tags in the result, zero by default which means no limit is set
<i>offset</i>	Number of tags to skip in the beginning of the result, zero by default
<i>order</i>	Allows to specify particular ordering of tags in the result, <code>NoOrder</code> by default
<i>orderDirection</i>	Specifies the direction of ordering, by default ascending direction is used;

**Returns**

The list of found tags per note

**5.41.3.40 `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

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

## 5.41.3.41 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

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

**5.41.3.42 listNotes()**

```
QList< Note > quantier::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**

<i>flag</i>	Input parameter used to set the filter for the desired notes to be listed
<i>options</i>	Options specifying which optionally includable fields of the note should actually be included
<i>errorDescription</i>	Error description if notes within the account could not be listed; if no error happens, this parameter is untouched
<i>limit</i>	Limit for the max number of notes in the result, zero by default which means no limit is set
<i>offset</i>	Number of notes to skip in the beginning of the result, zero by default
<i>order</i>	Allows to specify particular ordering of notes in the result, NoOrder by default
<i>orderDirection</i>	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder
<i>linkedNotebookGuid</i>	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

**5.41.3.43 listNotesByLocalUids()**

```
QList< Note > quantier::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

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

#### 5.41.3.44 listNotesPerNotebook()

```

QList< Note > quantier::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

<i>notebook</i>	<a href="#">Notebook</a> 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
<i>options</i>	Options specifying which optionally includable fields of the note should actually be included
<i>errorDescription</i>	Error description in case notes could not be listed
<i>flag</i>	Input parameter used to set the filter for the desired notes to be listed

## Parameters

<i>limit</i>	Limit for the max number of notes in the result, zero by default which means no limit is set
<i>offset</i>	Number of notes to skip in the beginning of the result, zero by default
<i>order</i>	Allows to specify particular ordering of notes in the result, NoOrder by default
<i>orderDirection</i>	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

## 5.41.3.45 listNotesPerNotebooksAndTags()

```
QList< Note > quantier::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

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

### 5.41.3.46 listNotesPerTag()

```
QList< Note > quantier::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

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

### 5.41.3.47 listSavedSearches()

```
QList< SavedSearch > quantier::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

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

**5.41.3.48 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**

<i>notebookGuid</i>	Remote Evernote service's guid of the notebook for which the shared notebooks are requested
<i>errorDescription</i>	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

**5.41.3.49 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**

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

## Parameters

<i>orderDirection</i>	Specifies the direction of ordering, by default ascending direction is used; this parameter has no meaning if order is equal to NoOrder
<i>linkedNotebookGuid</i>	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

## 5.41.3.50 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

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

**5.41.3.51 `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**

<i>errorDescription</i>	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

**5.41.3.52 `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**

<i>errorDescription</i>	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

#### 5.41.3.53 notebookCount()

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

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

##### Parameters

<i>errorDescription</i>	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

#### 5.41.3.54 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

<i>errorDescription</i>	Error description if the number of notes could not be returned
<i>options</i>	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

#### 5.41.3.55 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

<i>notebook</i>	<a href="#">Notebook</a> 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
<i>errorDescription</i>	Error description if the number of notes per given notebook could not be returned
<i>options</i>	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

**5.41.3.56 `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

<i>notebookLocalUids</i>	The list of notebook local uids used for filtering
<i>tagLocalUids</i>	The list of tag local uids used for filtering
<i>errorDescription</i>	Error description if the number of notes per notebooks and tags could not be returned
<i>options</i>	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

**5.41.3.57 `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

<i>tag</i>	<a href="#">Tag</a> 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
<i>errorDescription</i>	Error description if the number of notes per given tag could not be returned
<i>options</i>	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

**5.41.3.58 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

<i>noteCountsPerTagLocalUid</i>	The result hash: note counts by tag local uids
<i>errorDescription</i>	Error description if the number of notes per all tags could not be returned
<i>options</i>	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

**5.41.3.59 requiredLocalStoragePatches()**

```
QVector< std::shared_ptr< ILocalStoragePatch > > quentier::LocalStorageManager::required←
LocalStoragePatches ( )
```

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

### 5.41.3.60 savedSearchCount()

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

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

#### Parameters

<i>errorDescription</i>	Error description if the number of saved searhes could not be returned
-------------------------	--

#### Returns

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

### 5.41.3.61 switchUser()

```
void quantier::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

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

### 5.41.3.62 tagCount()

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

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

#### Parameters

<i>errorDescription</i>	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

**5.41.3.63 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**

<i>resource</i>	<a href="#">Resource</a> 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
<i>errorDescription</i>	Error description if resource could not be updated

**Returns**

True if resource was updated successfully, false otherwise

**5.41.3.64 updateLinkedNotebook()**

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

updateLinkedNotebook updates passed in [LinkedNotebook](#) in the local storage database; [LinkedNotebook](#) must have "remote" Evernote service's guid set.

**Parameters**

<i>linkedNotebook</i>	<a href="#">LinkedNotebook</a> to be updated in the local storage database
<i>errorDescription</i>	Error description if linked notebook could not be updated

**Returns**

True if linked notebook was updated successfully, false otherwise

## 5.41.3.65 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

<i>note</i>	<a href="#">Note</a> 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
<i>options</i>	Options specifying which optionally updatable fields of the note should actually be updated
<i>errorDescription</i>	Error description if note could not be updated

## Returns

True if note was updated successfully, false otherwise

## 5.41.3.66 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**

<i>notebook</i>	<a href="#">Notebook</a> 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)
<i>errorDescription</i>	Error description if the notebook could not be updated

**Returns**

True if the notebook was updated successfully, false otherwise

**5.41.3.67 updateSavedSearch()**

```
bool quantier::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**

<i>search</i>	<a href="#">SavedSearch</a> 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
<i>errorDescription</i>	Error description if <a href="#">SavedSearch</a> could not be updated

**Returns**

True if [SavedSearch](#) was updated successfully, false otherwise

**5.41.3.68 updateTag()**

```
bool quantier::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

<i>tag</i>	<a href="#">Tag</a> filled with values to be updated in the local storage database. <a href="#">Note</a> that it can be changed * as a result of the call: automatically filled with local uid if it was empty before the call
<i>errorDescription</i>	Error description if tag could not be updated

## Returns

True if tag was updated successfully, false otherwise

**5.41.3.69 updateUser()**

```
bool quantier::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

<i>user</i>	The user to be updated in the local storage database
<i>errorDescription</i>	Error description if the user could not be updated

## Returns

True if the user was updated successfully, false otherwise

**5.41.3.70 upgradeProgress**

```
void quantier::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

<i>progress</i>	The value from 0 to 1 denoting the database upgrade progress
-----------------	--

### 5.41.3.71 userCount()

```
int quantier::LocalStorageManager::userCount (
    QString & errorDescription ) const
```

userCount returns the number of non-deleted users currently stored in the local storage database

#### Parameters

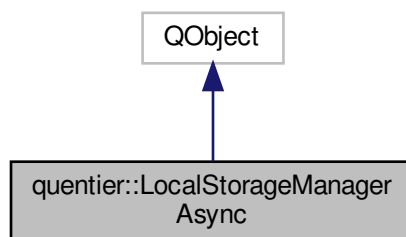
<i>errorDescription</i>	Error description if the number of users could not be returned
-------------------------	--

#### Returns

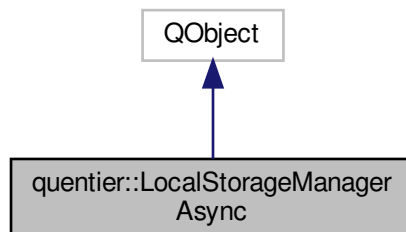
Either non-negative value with the number of users or -1 which means some error has occurred

## 5.42 quantier::LocalStorageManagerAsync Class Reference

Inheritance diagram for quantier::LocalStorageManagerAsync:



Collaboration diagram for quantier::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 >](#) noteCountsPerTagLocalUids, [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())

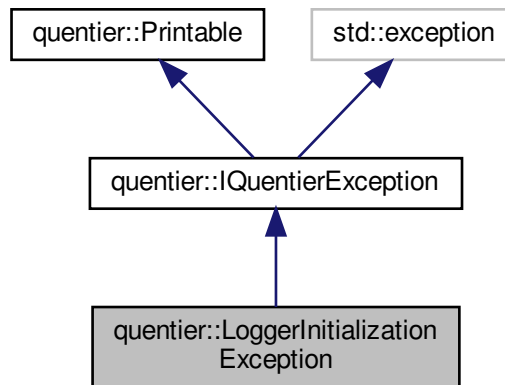
- void **listTagsFailed** (LocalStorageManager::ListObjectsOptions flag, size\_t limit, size\_t offset, [LocalStorageManager::ListTagsOrder](#) order, [LocalStorageManager::OrderDirection](#) orderDirection, QString linkedNotebookGuid, [ErrorString](#) errorDescription, QUuid requestId)
- void **listTagsWithNoteLocalUidsComplete** (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

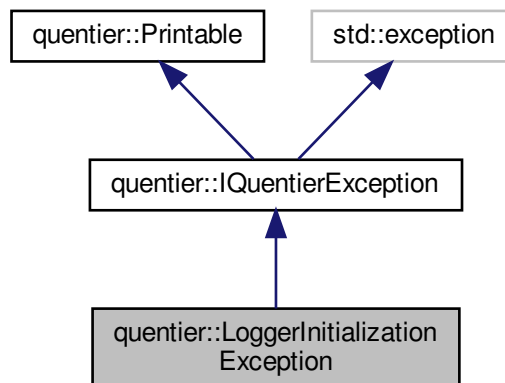
- **LocalStorageManagerAsync** (const [Account](#) &account, LocalStorageManager::StartupOptions options={}, QObject \*parent=nullptr)
- void **setUseCache** (const bool useCache)
- const [LocalStorageCacheManager](#) \* **localStorageCacheManager** () const
- bool **installCacheExpiryFunction** (const [ILocalStorageCacheExpiryChecker](#) &checker)
- const [LocalStorageManager](#) \* **localStorageManager** () const
- [LocalStorageManager](#) \* **localStorageManager** ()

## 5.43 quantier::LoggerInitializationException Class Reference

Inheritance diagram for quantier::LoggerInitializationException:



Collaboration diagram for quantier::LoggerInitializationException:



### Public Member Functions

- **LoggerInitializationException** (const [ErrorString](#) &message)

### Protected Member Functions

- const QString [exceptionDisplayName](#) () const override

### 5.43.1 Member Function Documentation

#### 5.43.1.1 exceptionDisplayName()

```
const QString quentier::LoggerInitializationException::exceptionDisplayName ( ) const [override],
[protected], [virtual]
```

Implements [quentier::IQuentierException](#).

## 5.44 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

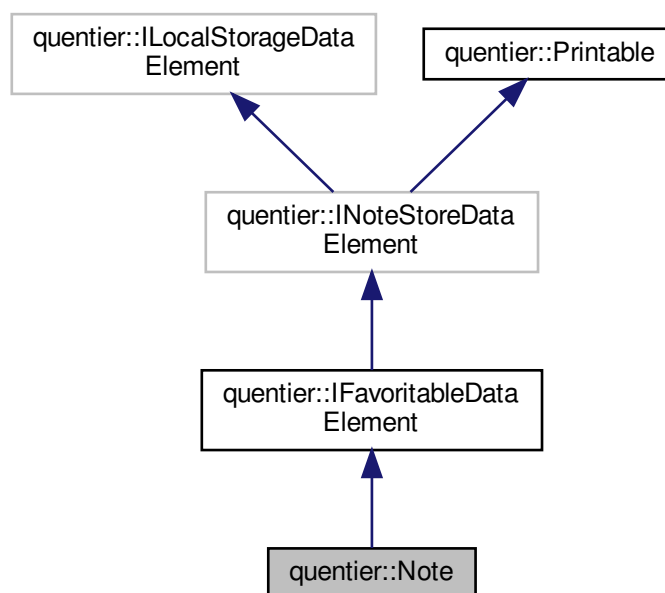
### Public Member Functions

- **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)

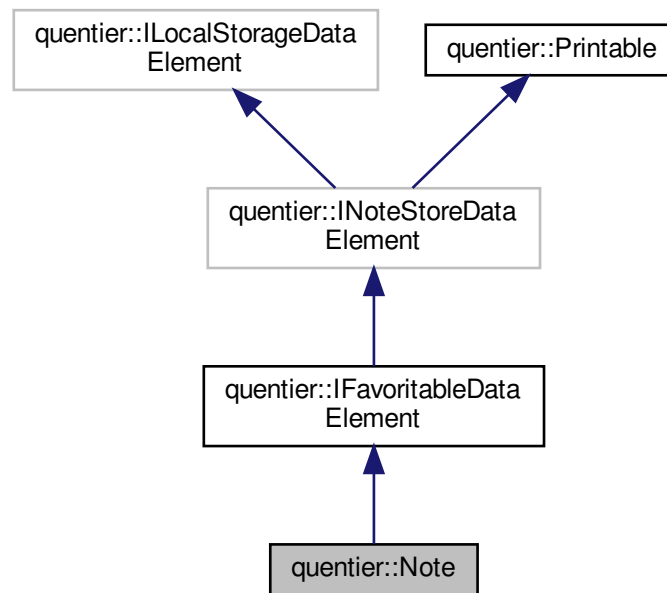


## 5.45 quantier::Note Class Reference

Inheritance diagram for quantier::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
- quint32 **contentLength** () const
- void **setContentLength** (const quint32 length)
- bool **hasCreationTimestamp** () const
- quint64 **creationTimestamp** () const
- void **setCreationTimestamp** (const quint64 timestamp)
- bool **hasModificationTimestamp** () const
- quint64 **modificationTimestamp** () const
- void **setModificationTimestamp** (const quint64 timestamp)
- bool **hasDeletionTimestamp** () const
- quint64 **deletionTimestamp** () const
- void **setDeletionTimestamp** (const quint64 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

### 5.45.1 Member Function Documentation

#### 5.45.1.1 checkParameters()

```
virtual bool qentier::Note::checkParameters (
    ErrorString & errorDescription ) const [override], [virtual]
```

Implements [qentier::INoteStoreDataElement](#).

#### 5.45.1.2 clear()

```
virtual void qentier::Note::clear ( ) [override], [virtual]
```

Implements [qentier::INoteStoreDataElement](#).

#### 5.45.1.3 guid()

```
virtual const QString & qentier::Note::guid ( ) const [override], [virtual]
```

Implements [qentier::INoteStoreDataElement](#).

#### 5.45.1.4 `hasGuid()`

```
virtual bool quentier::Note::hasGuid ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.45.1.5 `hasUpdateSequenceNumber()`

```
virtual bool quentier::Note::hasUpdateSequenceNumber ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.45.1.6 `print()`

```
virtual QTextStream & quentier::Note::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [quentier::Printable](#).

#### 5.45.1.7 `setGuid()`

```
virtual void quentier::Note::setGuid (
    const QString & guid ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.45.1.8 `setUpdateSequenceNumber()`

```
virtual void quentier::Note::setUpdateSequenceNumber (
    const quint32 usn ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

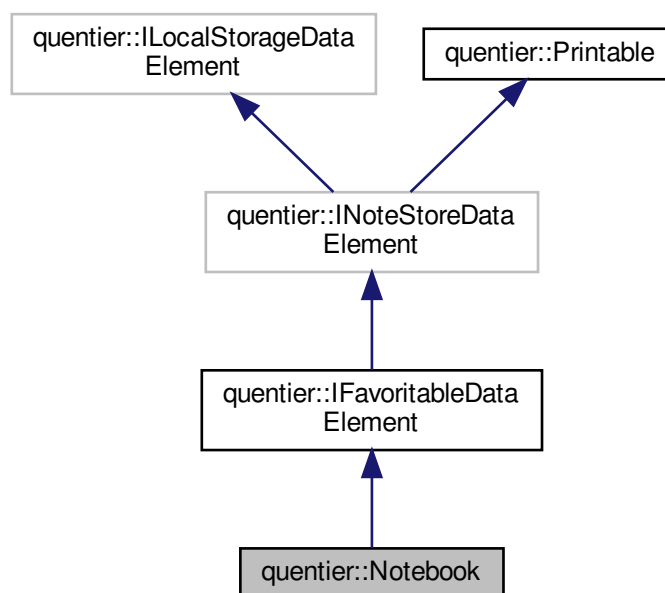
#### 5.45.1.9 `updateSequenceNumber()`

```
virtual quint32 quentier::Note::updateSequenceNumber ( ) const [override], [virtual]
```

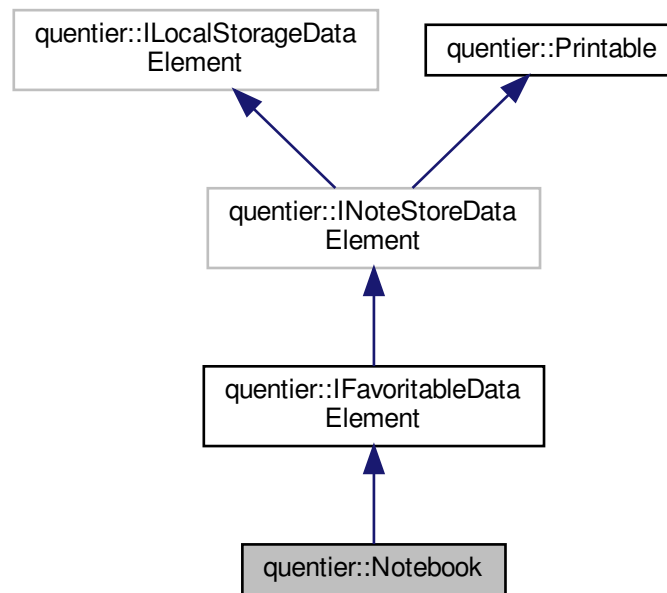
Implements [quentier::INoteStoreDataElement](#).

## 5.46 quantier::Notebook Class Reference

Inheritance diagram for quantier::Notebook:



Collaboration diagram for quantier::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

### 5.46.1 Member Function Documentation

#### 5.46.1.1 checkParameters()

```
virtual bool quantier::Notebook::checkParameters (
    ErrorString & errorDescription ) const [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.46.1.2 clear()

```
virtual void quantier::Notebook::clear ( ) [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.46.1.3 guid()

```
virtual const QString & quantier::Notebook::guid ( ) const [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.46.1.4 hasGuid()

```
virtual bool quantier::Notebook::hasGuid ( ) const [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.46.1.5 hasUpdateSequenceNumber()

```
virtual bool quantier::Notebook::hasUpdateSequenceNumber ( ) const [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.46.1.6 print()

```
virtual QTextStream & quantier::Notebook::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [quantier::Printable](#).

#### 5.46.1.7 setGuid()

```
virtual void quantier::Notebook::setGuid (
    const QString & guid ) [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.46.1.8 setUpdateSequenceNumber()

```
virtual void quantier::Notebook::setUpdateSequenceNumber (
    const quint32 usn ) [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.46.1.9 updateSequenceNumber()

```
virtual quint32 quantier::Notebook::updateSequenceNumber ( ) const [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

## 5.47 quantier::ENMLConverter::NoteContentToHtmlExtraData Struct Reference

### Public Attributes

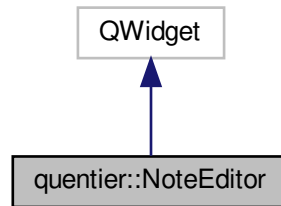
- quint64 **m\_numEnToDoNodes** = 0
- quint64 **m\_numHyperlinkNodes** = 0
- quint64 **m\_numEnCryptNodes** = 0
- quint64 **m\_numEnDecryptedNodes** = 0

## 5.48 quantier::NoteEditor Class Reference

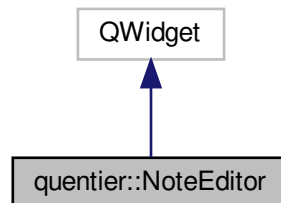
The [NoteEditor](#) class is a widget encapsulating all the functionality necessary for showing and editing notes.

```
#include <NoteEditor.h>
```

Inheritance diagram for quantier::NoteEditor:



Collaboration diagram for quantier::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

### 5.48.1 Detailed Description

The [NoteEditor](#) class is a widget encapsulating all the functionality necessary for showing and editing notes.

### 5.48.2 Member Function Documentation

#### 5.48.2.1 backend()

```
INoteEditorBackend * quantier::NoteEditor::backend ( )
```

#### Returns

the pointer to the note editor's backend

#### 5.48.2.2 clear()

```
void quantier::NoteEditor::clear ( )
```

Clear the contents of the note editor

#### 5.48.2.3 convertToNote

```
void quantier::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

#### 5.48.2.4 currentNoteLocalUid()

```
QString quantier::NoteEditor::currentNoteLocalUid ( ) const
```

Get the local uid of the note currently set to the note editor

#### 5.48.2.5 defaultFont()

```
const QFont * quantier::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

#### 5.48.2.6 defaultPalette()

```
QPalette quantier::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

#### 5.48.2.7 idleTime()

```
qint64 quantier::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



### 5.48.2.8 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.

### 5.48.2.9 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

<i>localStorageManager</i>	The reference to <a href="#">LocalStorageManagerAsync</a> , to set up signal-slot connections with it
<i>spellChecker</i>	The spell checker to be used by note editor for, well, spell-checking
<i>account</i>	Currently active account
<i>pBackgroundJobsThread</i>	Pointer to the thread to be used for scheduling of background jobs of <a href="#">NoteEditor</a> ; if null, <a href="#">NoteEditor</a> 's background jobs would take place in GUI thread

### 5.48.2.10 isEditorPageModified()

```
bool quentier::NoteEditor::isEditorPageModified ( ) const
```

#### Returns

true if there's content within the editor not yet converted to note, false otherwise

#### 5.48.2.11 isModified()

```
bool quantier::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

#### 5.48.2.12 isNoteLoaded()

```
bool quantier::NoteEditor::isNoteLoaded ( ) const
```

##### Returns

true if the note last set to the editor has been fully loaded already, false otherwise

#### 5.48.2.13 saveNoteToLocalStorage

```
void quantier::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

#### 5.48.2.14 setAccount()

```
void quantier::NoteEditor::setAccount (
    const Account & account )
```

Set the current account to the note editor

#### 5.48.2.15 setBackend()

```
void quantier::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

#### 5.48.2.16 setCurrentNoteLocalUid()

```
void quantier::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

<i>noteLocalUid</i>	The local uid of note
---------------------	-----------------------

**5.48.2.17 setDefaultFont**

```
void quantier::NoteEditor::setDefaultFont (
    const QFont & font ) [slot]
```

Sets the font which would be used by the editor by default

## Parameters

<i>font</i>	The font to be used by the editor by default
-------------	--

**5.48.2.18 setDefaultPalette**

```
void quantier::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

<i>pal</i>	The palette to be set. Invalid colors from it are substituted by colors from widget's palette by the editor
------------	---

**5.48.2.19 setFocus()**

```
void quantier::NoteEditor::setFocus ( )
```

Sets the focus to the backend note editor widget

#### 5.48.2.20 setInitialPageHtml()

```
void quentier::NoteEditor::setInitialPageHtml (
    const QString & html )
```

Set the html to be displayed when the note is not set to the editor

#### 5.48.2.21 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)

#### 5.48.2.22 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

#### 5.48.2.23 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

#### 5.48.2.24 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

<i>noteTitle</i>	The title of the note
------------------	-----------------------

#### 5.48.2.25 setTagIds

```
void quentier::NoteEditor::setTagIds (
```

```
const QStringList & tagLocalUids,
const QStringList & tagGuids ) [slot]
```

Invoke this slot to set tag local uids and/or tag guides 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

<i>tagLocalUids</i>	The list of tag local uids for the note
<i>tagGuids</i>	The list of tag guides for the note

#### 5.48.2.26 setUndoStack()

```
void quantier::NoteEditor::setUndoStack (
    QUndoStack * pUndoStack )
```

Set the undo stack for the note editor to use

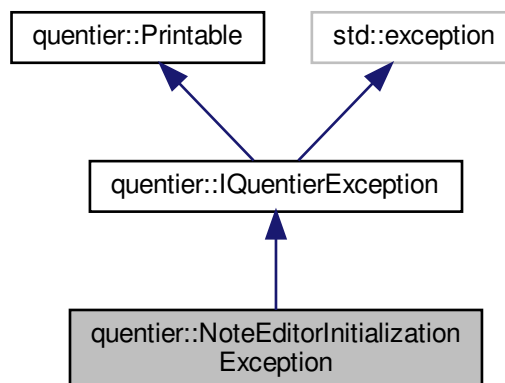
#### 5.48.2.27 undoStack()

```
const QUndoStack * quantier::NoteEditor::undoStack ( ) const
```

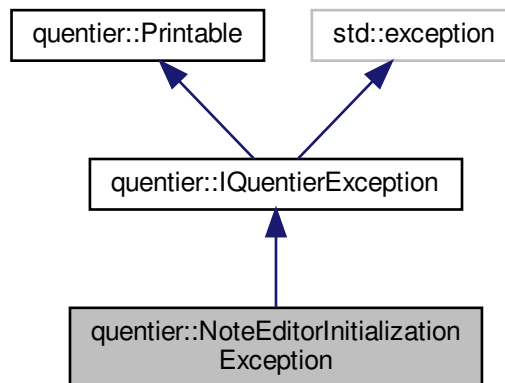
Get the undo stack serving to the note editor

## 5.49 quantier::NoteEditorInitializationException Class Reference

Inheritance diagram for quantier::NoteEditorInitializationException:



Collaboration diagram for `quentier::NoteEditorInitializationException`:



## Public Member Functions

- **NoteEditorInitializationException** (const [ErrorString](#) &message)

## Protected Member Functions

- virtual const `QString` [exceptionDisplayName](#) () const override

## 5.49.1 Member Function Documentation

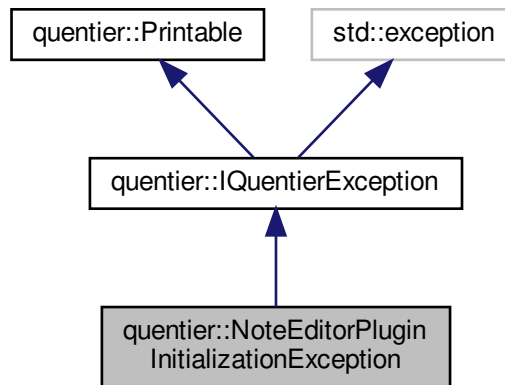
### 5.49.1.1 exceptionDisplayName()

```
virtual const QString quentier::NoteEditorInitializationException::exceptionDisplayName ( )
const [override], [protected], [virtual]
```

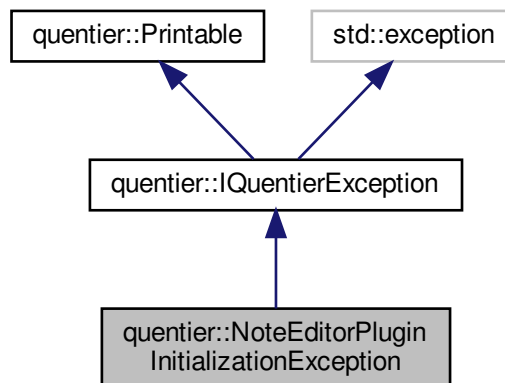
Implements [quentier::IQuentierException](#).

## 5.50 quantier::NoteEditorPluginInitializationException Class Reference

Inheritance diagram for quantier::NoteEditorPluginInitializationException:



Collaboration diagram for quantier::NoteEditorPluginInitializationException:



### Public Member Functions

- **NoteEditorPluginInitializationException** (const [ErrorString](#) &message)

### Protected Member Functions

- virtual const QString [exceptionDisplayName](#) () const override

### 5.50.1 Member Function Documentation

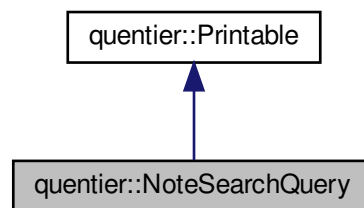
#### 5.50.1.1 exceptionDisplayName()

```
virtual const QString quentier::NoteEditorPluginInitializationException::exceptionDisplayName  
( ) const [override], [protected], [virtual]
```

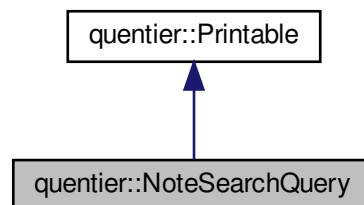
Implements [quentier::IQuentierException](#).

## 5.51 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

### 5.51.1 Member Function Documentation

### 5.51.1.1 `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

### 5.51.1.2 `print()`

```
virtual QTextStream & quentier::NoteSearchQuery::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [quentier::Printable](#).

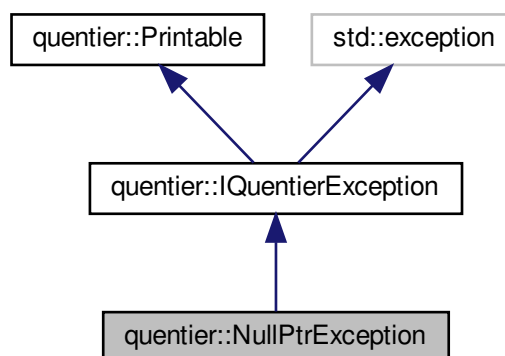
### 5.51.1.3 `queryString()`

```
const QString quentier::NoteSearchQuery::queryString ( ) const
```

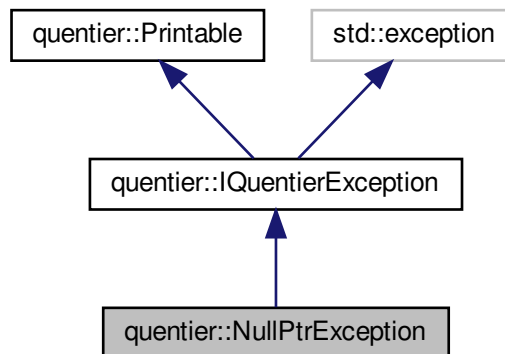
Returns the original non-parsed query string

## 5.52 `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

## 5.52.1 Member Function Documentation

### 5.52.1.1 `exceptionDisplayName()`

```
virtual const QString quentier::NullPtrException::exceptionDisplayName ( ) const [override],
[protected], [virtual]
```

Implements [quentier::IQuentierException](#).

## 5.53 `quentier::ResourceRecognitionIndexItem::ObjectItem` Struct Reference

## Public Member Functions

- bool `operator==` (const [ObjectItem](#) &other) const

## Public Attributes

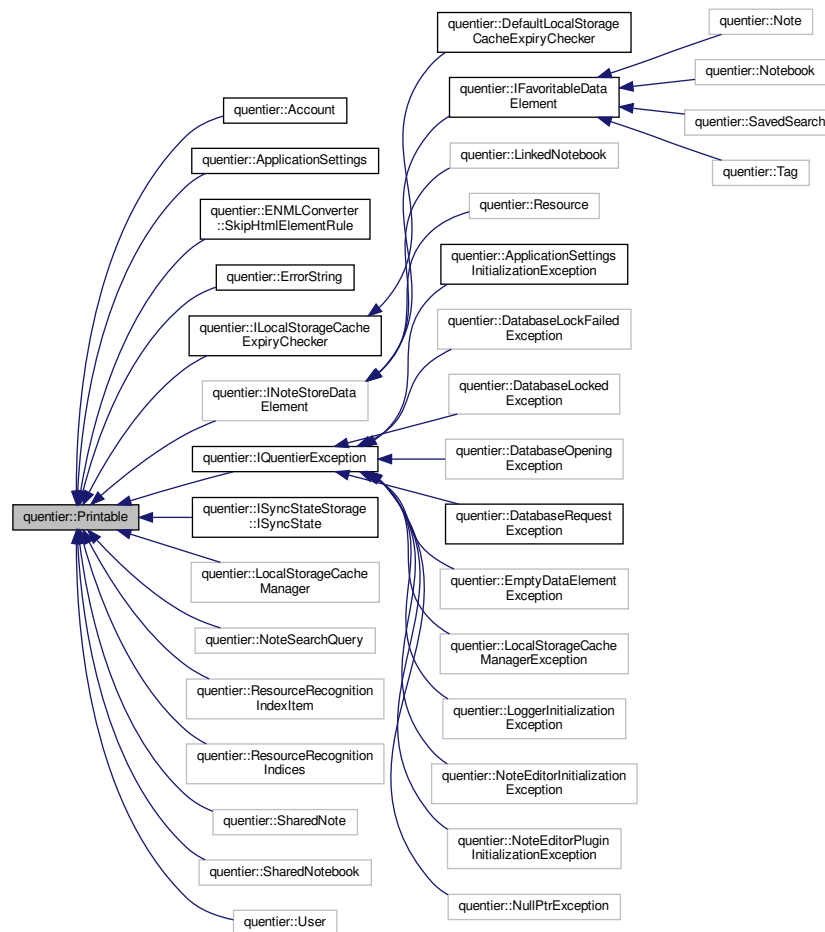
- QString **m\_objectType**
- int **m\_weight** = -1

## 5.54 quantier::Printable Class Reference

The [Printable](#) class is the interface for Quantier's internal classes which should be able to write themselves into QTextStream and/or convert to QString.

```
#include <Printable.h>
```

Inheritance diagram for quantier::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)

### 5.54.1 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.

### 5.54.2 Member Function Documentation

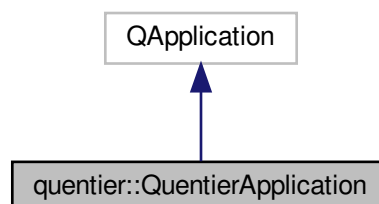
#### 5.54.2.1 print()

```
virtual QTextStream & quentier::Printable::print (
    QTextStream & strm ) const [pure virtual]
```

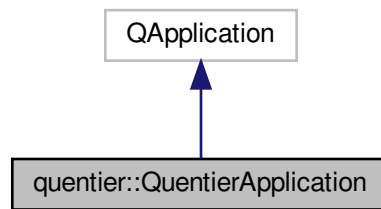
Implemented in [quentier::LocalStorageCacheExpiryChecker](#), and [quentier::DefaultLocalStorageCacheExpiryChecker](#).

## 5.55 quentier::QuentierApplication Class Reference

Inheritance diagram for quentier::QuentierApplication:



Collaboration diagram for quantier::QuantierApplication:



### Public Member Functions

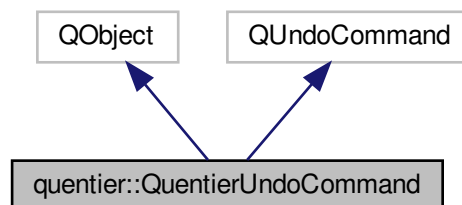
- **QuantierApplication** (int &argc, char \*argv[])
- virtual bool **notify** (QObject \*pObject, QEvent \*pEvent) override
- virtual bool **event** (QEvent \*pEvent) override

## 5.56 quantier::QuantierUndoCommand Class Reference

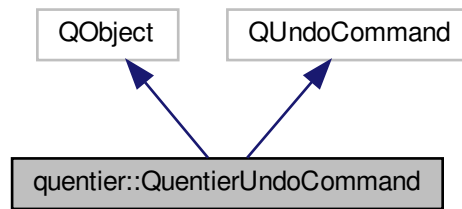
The [QuantierUndoCommand](#) class has the sole purpose of working around one quirky aspect of Qt's undo/redo framework: when you push `QUndoCommand` to `QUndoStack`, it calls "redo" method of that command. This class offers subclasses to implement their own methods for actual "undo" and "redo" commands while ignoring the attempts to "redo" anything if there were no previous "undo" call prior to that.

```
#include <QuantierUndoCommand.h>
```

Inheritance diagram for quantier::QuantierUndoCommand:



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 **undolmpl** ()=0
- virtual void **redolmpl** ()=0

### 5.56.1 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.

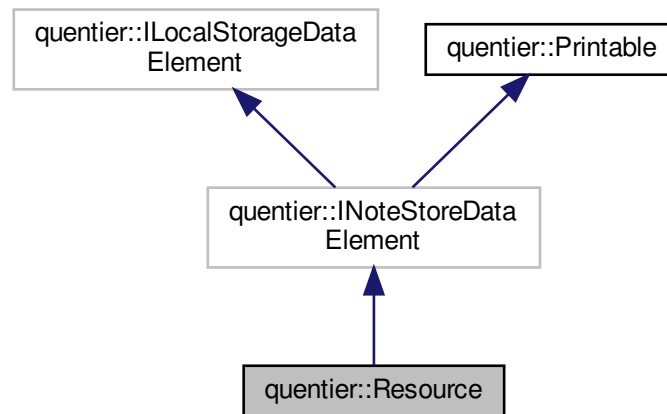
One thing which this design fails to see is the fact that the command may be already executed externally by the moment the QUndoCommand can be created. Suppose we can get the information about how to undo (and then again redo) that command. We create the corresponding QUndoCommand, set up the stuff for 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

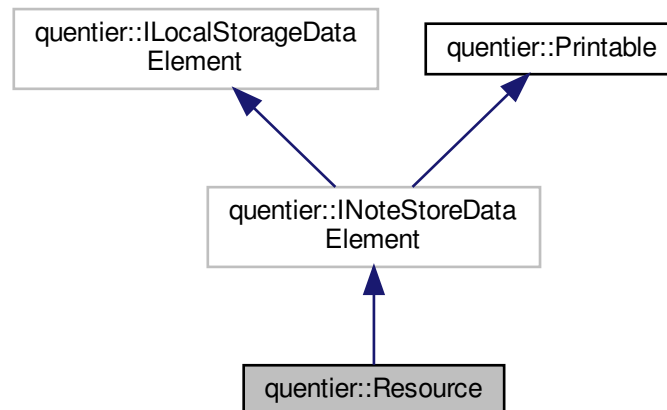


## 5.57 quantier::Resource Class Reference

Inheritance diagram for quantier::Resource:



Collaboration diagram for quantier::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

### 5.57.1 Member Function Documentation

#### 5.57.1.1 checkParameters()

```
virtual bool quantier::Resource::checkParameters (
    ErrorString & errorDescription ) const [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.57.1.2 clear()

```
virtual void quantier::Resource::clear ( ) [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.57.1.3 guid()

```
virtual const QString & quantier::Resource::guid ( ) const [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

#### 5.57.1.4 hasGuid()

```
virtual bool quentier::Resource::hasGuid ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.57.1.5 hasUpdateSequenceNumber()

```
virtual bool quentier::Resource::hasUpdateSequenceNumber ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.57.1.6 print()

```
virtual QTextStream & quentier::Resource::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [quentier::Printable](#).

#### 5.57.1.7 setGuid()

```
virtual void quentier::Resource::setGuid (
    const QString & guid ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.57.1.8 setUpdateSequenceNumber()

```
virtual void quentier::Resource::setUpdateSequenceNumber (
    const quint32 updateSequenceNumber ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

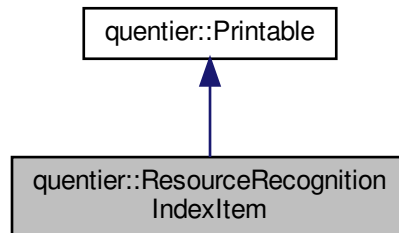
#### 5.57.1.9 updateSequenceNumber()

```
virtual quint32 quentier::Resource::updateSequenceNumber ( ) const [override], [virtual]
```

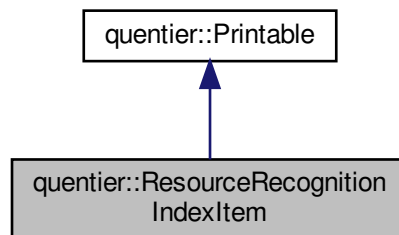
Implements [quentier::INoteStoreDataElement](#).

## 5.58 quantier::ResourceRecognitionIndexItem Class Reference

Inheritance diagram for quantier::ResourceRecognitionIndexItem:



Collaboration diagram for quantier::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< [ShapelItem](#) > **shapelItems** () const
- int **numShapelItems** () const
- bool **shapelItemAt** (const int shapelItemIndex, [ShapelItem](#) &shapelItem) const
- bool **setShapelItemAt** (const int shapelItemIndex, const [ShapelItem](#) &shapelItem)
- void **setShapelItems** (const QVector< [ShapelItem](#) > &shapelItems)
- void **reserveShapelItemsSpace** (const int numItems)
- void **addShapelItem** (const [ShapelItem](#) &item)
- bool **removeShapelItem** (const [ShapelItem](#) &item)
- bool **removeShapelItemAt** (const int shapelItemIndex)
- 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

### 5.58.1 Member Function Documentation

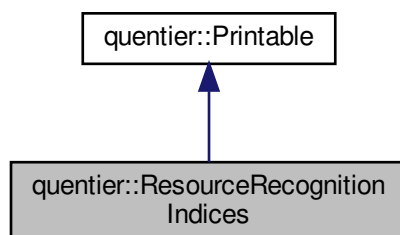
#### 5.58.1.1 `print()`

```
virtual QTextStream & quentier::ResourceRecognitionIndexItem::print (
    QTextStream & strm ) const [override], [virtual]
```

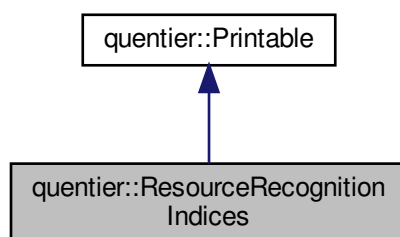
Implements [quentier::Printable](#).

## 5.59 `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

### 5.59.1 Member Function Documentation

#### 5.59.1.1 print()

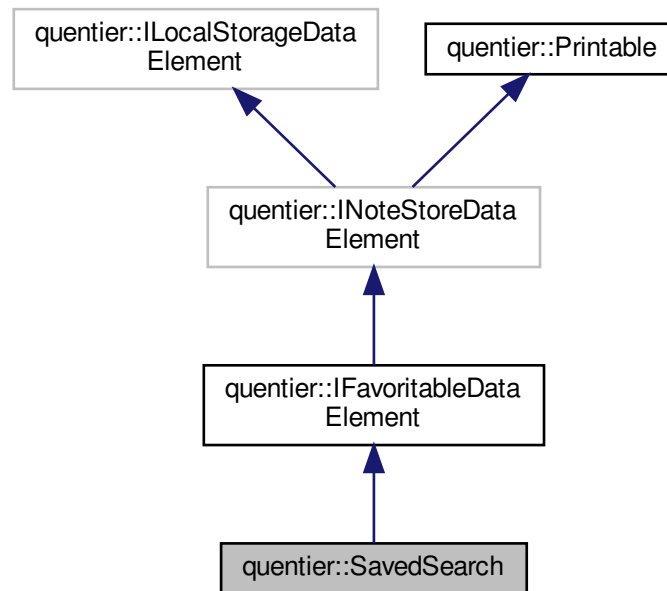
```
virtual QTextStream & quentier::ResourceRecognitionIndices::print (  
    QTextStream & strm ) const    [override], [virtual]
```

Implements [quentier::Printable](#).

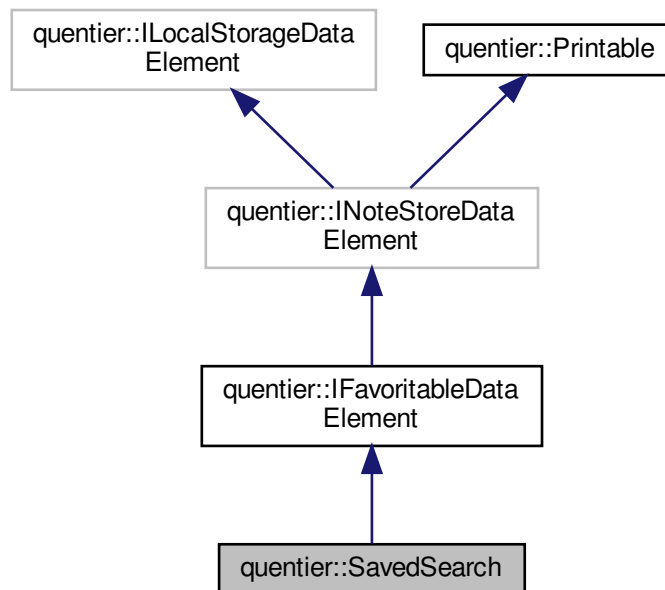


## 5.60 quantier::SavedSearch Class Reference

Inheritance diagram for quantier::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 quint8 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

### 5.60.1 Member Function Documentation

#### 5.60.1.1 checkParameters()

```
virtual bool quentier::SavedSearch::checkParameters (
    ErrorString & errorDescription ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.60.1.2 clear()

```
virtual void quentier::SavedSearch::clear ( ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.60.1.3 guid()

```
virtual const QString & quentier::SavedSearch::guid ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.60.1.4 hasGuid()

```
virtual bool quentier::SavedSearch::hasGuid ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.60.1.5 hasUpdateSequenceNumber()

```
virtual bool quentier::SavedSearch::hasUpdateSequenceNumber ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.60.1.6 print()

```
virtual QTextStream & quentier::SavedSearch::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [quentier::Printable](#).

#### 5.60.1.7 setGuid()

```
virtual void quentier::SavedSearch::setGuid (
    const QString & guid ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.60.1.8 setUpdateSequenceNumber()

```
virtual void quentier::SavedSearch::setUpdateSequenceNumber (
    const quint32 usn ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.60.1.9 updateSequenceNumber()

```
virtual qint32 quantier::SavedSearch::updateSequenceNumber ( ) const [override], [virtual]
```

Implements [quantier::INoteStoreDataElement](#).

## 5.61 quantier::ResourceRecognitionIndexItem::Shapeltem Struct Reference

### Public Member Functions

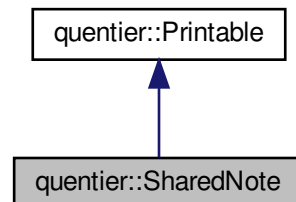
- `bool operator== (const Shapeltem &other) const`

### Public Attributes

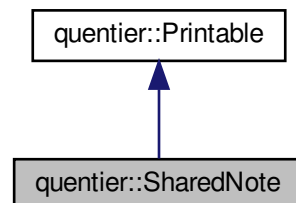
- `QString m_shapeType`
- `int m_weight = -1`

## 5.62 quantier::SharedNote Class Reference

Inheritance diagram for `quantier::SharedNote`:



Collaboration diagram for `quantier::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
- quint32 **sharerUserId** () const
- void **setSharerUserId** (const quint32 id)
- bool **hasRecipientIdentityId** () const
- quint64 **recipientIdentityId** () const
- void **setRecipientIdentityId** (const quint64 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 quint32 recipientIdentityContactType)
- bool **hasRecipientIdentityContactPhotoUrl** () const
- const QString & **recipientIdentityContactPhotoUrl** () const
- void **setRecipientIdentityContactPhotoUrl** (const QString &recipientIdentityPhotoUrl)
- bool **hasRecipientIdentityContactPhotoLastUpdated** () const
- quint64 **recipientIdentityContactPhotoLastUpdated** () const
- void **setRecipientIdentityContactPhotoLastUpdated** (const quint64 recipientIdentityPhotoLastUpdated)
- bool **hasRecipientIdentityContactMessagingPermit** () const
- const QByteArray & **recipientIdentityContactMessagingPermit** () const
- void **setRecipientIdentityContactMessagingPermit** (const QByteArray &messagingPermit)
- bool **hasRecipientIdentityContactMessagingPermitExpires** () const
- quint64 **recipientIdentityContactMessagingPermitExpires** () const
- void **setRecipientIdentityContactMessagingPermitExpires** (const quint64 timestamp)
- bool **hasRecipientIdentityUserId** () const
- quint32 **recipientIdentityUserId** () const
- void **setRecipientIdentityUserId** (const quint32 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

### 5.62.1 Member Function Documentation

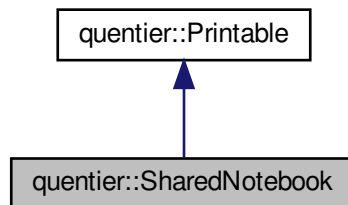
#### 5.62.1.1 print()

```
virtual QTextStream & quentier::SharedNote::print (
    QTextStream & strm ) const [override], [virtual]
```

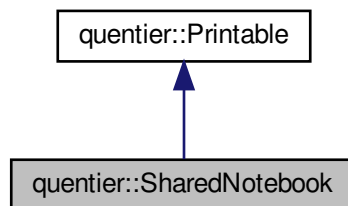
Implements [quentier::Printable](#).

## 5.63 quantier::SharedNotebook Class Reference

Inheritance diagram for quantier::SharedNotebook:



Collaboration diagram for quantier::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

### 5.63.1 Member Function Documentation

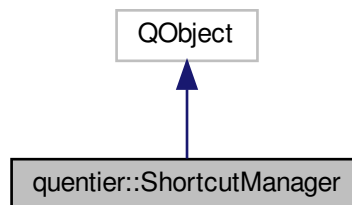
#### 5.63.1.1 `print()`

```
virtual QTextStream & quantier::SharedNotebook::print (  
    QTextStream & strm ) const  [override], [virtual]
```

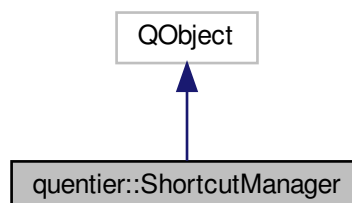
Implements [quantier::Printable](#).

## 5.64 `quantier::ShortcutManager` Class Reference

Inheritance diagram for `quantier::ShortcutManager`:



Collaboration diagram for `quantier::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

### 5.64.1 Member Function Documentation

#### 5.64.1.1 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

#### 5.64.1.2 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

#### 5.64.1.3 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)

#### 5.64.1.4 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)

## 5.64.1.5 userShortcut() [1/2]

```
QKeySequence quantier::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

## 5.64.1.6 userShortcut() [2/2]

```
QKeySequence quantier::ShortcutManager::userShortcut (
    const QString & nonStandardKey,
    const Account & account,
    const QString & context = {} ) const
```

## Returns

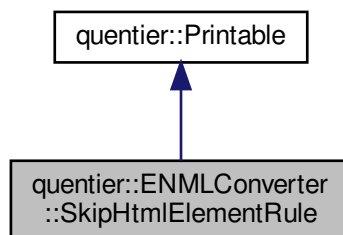
User defined shortcut for the non-standard key if present, otherwise empty key sequence

## 5.65 quantier::ENMLConverter::SkipHtmlElementRule Class Reference

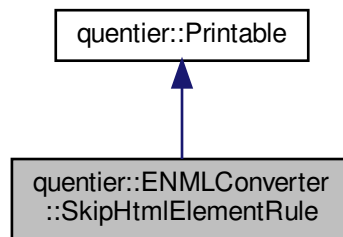
The [SkipHtmlElementRule](#) class describes the set of rules for HTML -> ENML conversion about the HTML elements that should not be actually converted to ENML due to their nature of being "helper" elements for the display or functioning of something within the note editor's page. The HTML -> ENML conversion 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 quantier::ENMLConverter::SkipHtmlElementRule:



Collaboration diagram for `quentier::ENMLConverter::SkipHtmlElementRule`:



## Public Types

- enum class **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

### 5.65.1 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.

## 5.65.2 Member Function Documentation

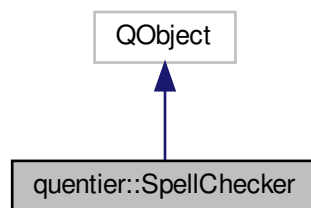
### 5.65.2.1 print()

```
virtual QTextStream & quentier::ENMLConverter::SkipHtmlElementRule::print (
    QTextStream & strm ) const [override], [virtual]
```

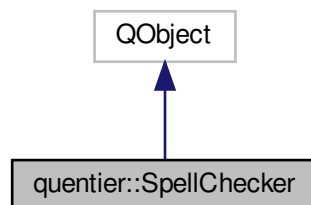
Implements [quentier::Printable](#).

## 5.66 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

## 5.67 quotient::StringUtils::StringFilterPredicate Struct Reference

### Public Member Functions

- **StringFilterPredicate** (QSet< QString > &filteredStrings)
- bool **operator()** (const QString &str) const

### Public Attributes

- QSet< QString > & **m\_filteredStrings**

## 5.68 quotient::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

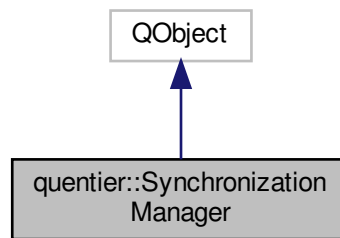


## 5.69 quantier::SynchronizationManager Class Reference

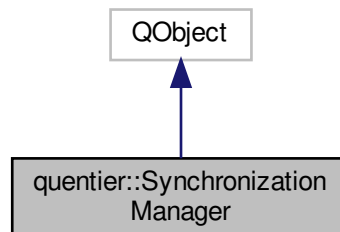
The [SynchronizationManager](#) class encapsulates methods and signals & slots required to perform the full or partial synchronization of data with remote Evernote servers. The class also deals with authentication with Evernote service through OAuth.

```
#include <SynchronizationManager.h>
```

Inheritance diagram for quantier::SynchronizationManager:



Collaboration diagram for quantier::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 [linkedNotebookSyncChunksDownloadProgress](#) (qint32 highestDownloadedUsn, qint32 highestServer↔Usn, qint32 lastPreviousUsn, [LinkedNotebook](#) linkedNotebook)
- void [linkedNotebooksSyncChunksDownloaded](#) ()
- void [notesDownloadProgress](#) (quint32 notesDownloaded, quint32 totalNotesToDownload)
- void [linkedNotebooksNotesDownloadProgress](#) (quint32 notesDownloaded, quint32 totalNotesToDownload)
- void [resourcesDownloadProgress](#) (quint32 resourcesDownloaded, quint32 totalResourcesToDownload)
- void [linkedNotebooksResourcesDownloadProgress](#) (quint32 resourcesDownloaded, quint32 total↔ResourcesToDownload)
- 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 pSync↔StateStorage={})
- bool [active](#) () const
- bool [downloadNoteThumbnailsOption](#) () const

### 5.69.1 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.

### 5.69.2 Constructor & Destructor Documentation

### 5.69.2.1 SynchronizationManager()

```
quentier::SynchronizationManager::SynchronizationManager (
    QString host,
    LocalStorageManagerAsync & localStorageManagerAsync,
    IAuthenticationManager & authenticationManager,
    QObject * parent = nullptr,
    INoteStorePtr pNoteStore = {},
    IUserStorePtr pUserStore = {},
    IKeychainServicePtr pKeychainService = {},
    ISyncStateStoragePtr pSyncStateStorage = {} )
```

#### Parameters

<i>host</i>	The host to use for the connection with the Evernote service - typically <a href="http://www.evernote.com">www.evernote.com</a> but could also be <a href="http://sandbox.evernote.com">sandbox.evernote.com</a> or some other one
<i>localStorageManagerAsync</i>	Local storage manager
<i>authenticationManager</i>	Reference to an object implementing <a href="#">IAuthenticationManager</a> interface; <a href="#">SynchronizationManager</a> does not store this reference, it only connects to the object via signals and slots during construction
<i>pNoteStore</i>	Pointer to an object implementing <a href="#">INoteStore</a> interface; if nullptr, <a href="#">SynchronizationManager</a> would create and use its own instance; otherwise <a href="#">SynchronizationManager</a> would set itself as the parent of the passed in object
<i>pUserStore</i>	Pointer to an object implementing <a href="#">IUserStore</a> interface; if nullptr, <a href="#">SynchronizationManager</a> would create and use its own instance
<i>pKeychainService</i>	Pointer to an object implementing <a href="#">IKeychainService</a> interface; if nullptr, <a href="#">SynchronizationManager</a> would create and use its own default instance; otherwise <a href="#">SynchronizationManager</a> would set itself as the parent of the passed in object
<i>pSyncStateStorage</i>	Pointer to an object implementing <a href="#">ISyncStateStorage</a> interface; if nullptr, <a href="#">SynchronizationManager</a> would create and use its own instance; otherwise <a href="#">SynchronizationManager</a> would set itself as the parent of the passed in object

## 5.69.3 Member Function Documentation

### 5.69.3.1 active()

```
bool quentier::SynchronizationManager::active ( ) const
```

#### Returns

True if the synchronization is being performed at the moment, false otherwise

### 5.69.3.2 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.

### 5.69.3.3 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.

### 5.69.3.4 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 invocation of authenticate slot.

#### Parameters

<i>success</i>	True if the authentication was successful, false otherwise
<i>errorDescription</i>	The textual explanation of the failure to authenticate the new user
<i>account</i>	The account of the authenticated user

### 5.69.3.5 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

<i>success</i>	True if the authentication was revoked successfully, false otherwise
<i>errorDescription</i>	The textual explanation of the failure to revoke the authentication
<i>userId</i>	The ID of the user for which the revoke of the authentication was requested

### 5.69.3.6 detectedConflictDuringLocalChangesSending

```
void quantier::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.

### 5.69.3.7 downloadNoteThumbnailsOption()

```
bool quantier::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

### 5.69.3.8 failed

```
void quantier::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

### 5.69.3.9 finished

```
void quantier::SynchronizationManager::finished (
    Account account,
    bool somethingDownloaded,
    bool somethingSent ) [signal]
```

This signal is emitted when the synchronization is finished

## Parameters

<i>account</i>	Represents the latest version of <a href="#">Account</a> structure filled during the synchronization procedure
<i>somethingDownloaded</i>	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
<i>somethingSent</i>	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

**5.69.3.10 linkedNotebooksNotesDownloadProgress**

```
void quantier::SynchronizationManager::linkedNotebooksNotesDownloadProgress (
    quint32 notesDownloaded,
    quint32 totalNotesToDownload ) [signal]
```

This signal is emitted on each successful download of full note data from linked notebooks.

## Parameters

<i>notesDownloaded</i>	The number of notes downloaded by the moment
<i>totalNotesToDownload</i>	The total number of notes that need to be downloaded

**5.69.3.11 linkedNotebooksResourcesDownloadProgress**

```
void quantier::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

<i>resourcesDownloaded</i>	The number of resources downloaded by the moment
<i>totalResourcesToDownload</i>	The total number of resources that need to be downloaded

**5.69.3.12 linkedNotebooksSyncChunksDownloaded**

```
void quantier::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

### 5.69.3.13 linkedNotebookSyncChunksDownloadProgress

```
void quantier::SynchronizationManager::linkedNotebookSyncChunksDownloadProgress (
    quint32 highestDownloadedUsn,
    quint32 highestServerUsn,
    quint32 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  $(\text{highestDownloadedUsn} - \text{lastPreviousUsn}) / (\text{highestServerUsn} - \text{lastPreviousUsn}) * 100\%$ . 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

<i>highestDownloadedUsn</i>	The highest update sequence number within data items from linked notebook sync chunks downloaded so far
<i>highestServerUsn</i>	The current highest update sequence number within the linked notebook
<i>lastPreviousUsn</i>	The last update sequence number from previous sync of the given linked notebook; if current sync is the first one, this value is zero
<i>linkedNotebook</i>	The linked notebook which sync chunks download progress is reported

### 5.69.3.14 notesDownloadProgress

```
void quantier::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

<i>notesDownloaded</i>	The number of notes downloaded by the moment
<i>totalNotesToDownload</i>	The total number of notes that need to be downloaded

### 5.69.3.15 preparedDirtyObjectsForSending

```
void quantier::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

### 5.69.3.16 preparedLinkedNotebooksDirtyObjectsForSending

```
void quantier::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

### 5.69.3.17 rateLimitExceeded

```
void quentier::SynchronizationManager::rateLimitExceeded (
    quint32 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

<i>secondsToWait</i>	The amount of time (in seconds) necessary to wait before the synchronization will continue
----------------------	--

### 5.69.3.18 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 algorithm switches to sending the local changes back to the Evernote service.

#### Parameters

<i>somethingDownloaded</i>	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
----------------------------	--

### 5.69.3.19 remoteToLocalSyncStopped

```
void quentier::SynchronizationManager::remoteToLocalSyncStopped ( ) [signal]
```

This signal is emitted when the "remote to local" synchronization step is stopped

### 5.69.3.20 resourcesDownloadProgress

```
void quentier::SynchronizationManager::resourcesDownloadProgress (
    quint32 resourcesDownloaded,
    quint32 totalResourcesToDownload ) [signal]
```

This signal is emitted on each successful download 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

<i>resourcesDownloaded</i>	The number of resources downloaded by the moment
<i>totalResourcesToDownload</i>	The total number of resources that need to be downloaded



### 5.69.3.21 revokeAuthentication

```
void quantier::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

### 5.69.3.22 sendLocalChangesStopped

```
void quantier::SynchronizationManager::sendLocalChangesStopped ( ) [signal]
```

This signal is emitted when the "send local changes" synchronization step is stopped

### 5.69.3.23 setAccount

```
void quantier::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

### 5.69.3.24 setAccountDone

```
void quantier::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

### 5.69.3.25 setDownloadInkNoteImages

```
void quantier::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

### 5.69.3.26 setDownloadInkNoteImagesDone

```
void quentier::SynchronizationManager::setDownloadInkNoteImagesDone (
    bool flag ) [signal]
```

This signal is emitted in response to invoking the setDownloadInkNoteImages slot after the setting is accepted

### 5.69.3.27 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

### 5.69.3.28 setDownloadNoteThumbnailsDone

```
void quentier::SynchronizationManager::setDownloadNoteThumbnailsDone (
    bool flag ) [signal]
```

This signal is emitted in response to invoking the setDownloadNoteThumbnails slot after the setting is accepted

### 5.69.3.29 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 applicationPersistentStoragePath 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

### 5.69.3.30 setInkNoteImagesStoragePathDone

```
void quentier::SynchronizationManager::setInkNoteImagesStoragePathDone (
    QString path ) [signal]
```

This signal is emitted in response to invoking the setInkNoteImagesStoragePath slot after the setting is accepted

### 5.69.3.31 started

```
void quantier::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)

### 5.69.3.32 stop

```
void quantier::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

### 5.69.3.33 stopped

```
void quantier::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)

### 5.69.3.34 syncChunksDownloaded

```
void quantier::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

### 5.69.3.35 syncChunksDownloadProgress

```
void quantier::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  $(\text{highestDownloadedUsn} - \text{lastPreviousUsn}) / (\text{highestServerUsn} - \text{lastPreviousUsn}) * 100\%$

#### Parameters

<i>highestDownloadedUsn</i>	The highest update sequence number within data items from sync chunks downloaded so far
<i>highestServerUsn</i>	The current highest update sequence number within the account
<i>lastPreviousUsn</i>	The last update sequence number from previous sync; if current sync is the first one, this value is zero

### 5.69.3.36 synchronize

```
void quantier::SynchronizationManager::synchronize ( ) [slot]
```

Use this slot to launch the synchronization of data

### 5.69.3.37 willRepeatRemoteToLocalSyncAfterSendingChanges

```
void quantier::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.

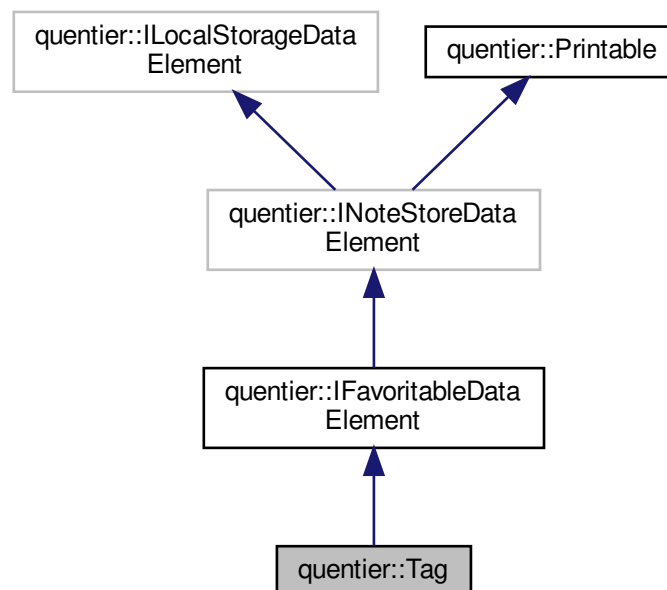
## 5.70 quantier::SysInfo Class Reference

### Public Member Functions

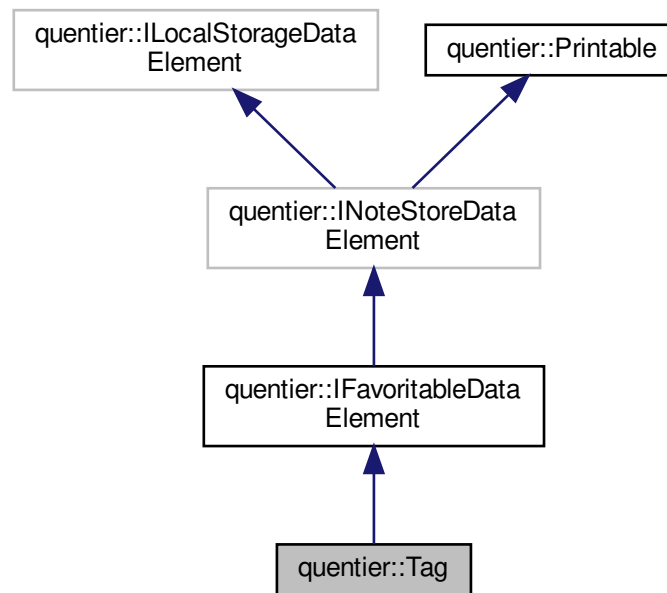
- quint64 **pageSize** ()
- quint64 **totalMemory** ()
- quint64 **freeMemory** ()
- QString **stackTrace** ()
- QString **platformName** ()

## 5.71 quantier::Tag Class Reference

Inheritance diagram for quantier::Tag:



Collaboration diagram for quantier::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

### 5.71.1 Member Function Documentation

#### 5.71.1.1 **checkParameters()**

```
virtual bool quentier::Tag::checkParameters (
    ErrorString & errorDescription ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.71.1.2 **clear()**

```
virtual void quentier::Tag::clear ( ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.71.1.3 **guid()**

```
virtual const QString & quentier::Tag::guid ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.71.1.4 **hasGuid()**

```
virtual bool quentier::Tag::hasGuid ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.71.1.5 `hasUpdateSequenceNumber()`

```
virtual bool quentier::Tag::hasUpdateSequenceNumber ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.71.1.6 `print()`

```
virtual QTextStream & quentier::Tag::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [quentier::Printable](#).

#### 5.71.1.7 `setGuid()`

```
virtual void quentier::Tag::setGuid (
    const QString & guid ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.71.1.8 `setUpdateSequenceNumber()`

```
virtual void quentier::Tag::setUpdateSequenceNumber (
    const quint32 usn ) [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

#### 5.71.1.9 `updateSequenceNumber()`

```
virtual quint32 quentier::Tag::updateSequenceNumber ( ) const [override], [virtual]
```

Implements [quentier::INoteStoreDataElement](#).

## 5.72 `quentier::ResourceRecognitionIndexItem::TextItem` Struct Reference

### Public Member Functions

- `bool operator==` (const [TextItem](#) &*other*) const

## Public Attributes

- QString **m\_text**
- int **m\_weight** = -1

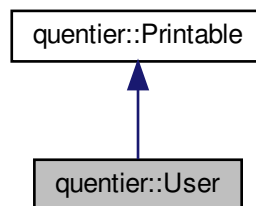
## 5.73 quantier::UidGenerator Class Reference

### Static Public Member Functions

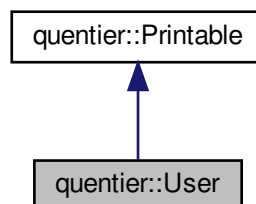
- static QString **Generate** ()
- static QString **UidToString** (const QUuid &uid)

## 5.74 quantier::User Class Reference

Inheritance diagram for quantier::User:



Collaboration diagram for quantier::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

### 5.74.1 Member Function Documentation

#### 5.74.1.1 print()

```
virtual QTextStream & qeventier::User::print (
    QTextStream & strm ) const [override], [virtual]
```

Implements [qeventier::Printable](#).

## Chapter 6

# File Documentation

### 6.1 DecryptedTextManager.h

```
1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_ENML_DECRYPTED_TEXT_MANAGER_H
20 #define LIB_QUENTIER_ENML_DECRYPTED_TEXT_MANAGER_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QtGlobal>
25
26 namespace quentier {
27
28 QT_FORWARD_DECLARE_CLASS(DecryptedTextManagerPrivate)
29
30 class QUENTIER_EXPORT DecryptedTextManager
31 {
32 public:
33     DecryptedTextManager();
34     virtual ~DecryptedTextManager();
35
36     void addEntry(
37         const QString & hash, const QString & decryptedText,
38         const bool rememberForSession, const QString & passphrase,
39         const QString & cipher, const size_t keyLength);
40
41     void removeEntry(const QString & hash);
42
43     void clearNonRememberedForSessionEntries();
44
45     bool findDecryptedTextByEncryptedText(
46         const QString & encryptedText, QString & decryptedText,
47         bool & rememberForSession) const;
48
49     bool modifyDecryptedText(
50         const QString & originalEncryptedText, const QString & newDecryptedText,
51         QString & newEncryptedText);
52
53 private:
54     Q_DISABLE_COPY(DecryptedTextManager)
55
56     DecryptedTextManagerPrivate * const d_ptr;
57     Q_DECLARE_PRIVATE(DecryptedTextManager)
58 };
```

```

59
60 } // namespace quantier
61
62 #endif // LIB_QUENTIER_ENML_DECRYPTED_TEXT_MANAGER_H

```

## 6.2 ENMLConverter.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_ENML_ENML_CONVERTER_H
20 #define LIB_QUENTIER_ENML_ENML_CONVERTER_H
21
22 #include <quantier/types/ErrorString.h>
23 #include <quantier/types/Note.h>
24 #include <quantier/utility/Linkage.h>
25 #include <quantier/utility/Printable.h>
26
27 #include <QHash>
28 #include <QSet>
29 #include <QString>
30 #include <QTextDocument>
31
32 namespace quantier {
33
34 QT_FORWARD_DECLARE_CLASS(DecryptedTextManager)
35 QT_FORWARD_DECLARE_CLASS(ENMLConverterPrivate)
36 QT_FORWARD_DECLARE_CLASS(Resource)
37
38
39 class QUENTIER_EXPORT ENMLConverter
40 {
41 public:
42     ENMLConverter();
43
44     virtual ~ENMLConverter();
45
46     class QUENTIER_EXPORT SkipHtmlElementRule : public Printable
47     {
48     public:
49         enum class ComparisonRule
50         {
51             Equals = 0,
52             StartsWith,
53             EndsWith,
54             Contains
55         };
56
57         friend QUENTIER_EXPORT QTextStream & operator<<(
58             QTextStream & strm, const ComparisonRule rule);
59
60         virtual QTextStream & print(QTextStream & strm) const override;
61
62         QString m_elementNameToSkip;
63         ComparisonRule m_elementNameComparisonRule = ComparisonRule::Equals;
64         Qt::CaseSensitivity m_elementNameCaseSensitivity = Qt::CaseSensitive;
65
66         QString m_attributeNameToSkip;
67         ComparisonRule m_attributeNameComparisonRule = ComparisonRule::Equals;
68         Qt::CaseSensitivity m_attributeNameCaseSensitivity = Qt::CaseSensitive;
69
70         QString m_attributeValueToSkip;
71         ComparisonRule m_attributeValueComparisonRule = ComparisonRule::Equals;
72         Qt::CaseSensitivity m_attributeValueCaseSensitivity = Qt::CaseSensitive;
73
74         bool m_includeElementContents = false;
75     };
76 };
77
78
79
80
81
82
83
84
85
86
87
88
89
90

```

```

91     bool htmlToNoteContent(
92         const QString & html, QString & noteContent,
93         DecryptedTextManager & decryptedTextManager,
94         ErrorString & errorDescription,
95         const QVector<SkipHtmlElementRule> & skipRules = {}) const;
96
110    bool cleanupExternalHtml(
111        const QString & inputHtml, QString & cleanedUpHtml,
112        ErrorString & errorDescription) const;
113
127    bool htmlToQTextDocument(
128        const QString & html, QTextDocument & doc,
129        ErrorString & errorDescription,
130        const QVector<SkipHtmlElementRule> & skipRules = {}) const;
131
132    struct NoteContentToHtmlExtraData
133    {
134        quint64 m_numEnToDoNodes = 0;
135        quint64 m_numHyperlinkNodes = 0;
136        quint64 m_numEnCryptNodes = 0;
137        quint64 m_numEnDecryptedNodes = 0;
138    };
139
140    bool noteContentToHtml(
141        const QString & noteContent, QString & html,
142        ErrorString & errorDescription,
143        DecryptedTextManager & decryptedTextManager,
144        NoteContentToHtmlExtraData & extraData) const;
145
146    bool validateEnml(
147        const QString & enml, ErrorString & errorDescription) const;
148
149    bool validateAndFixupEnml(
150        QString & enml, ErrorString & errorDescription) const;
151
152    static bool noteContentToPlainText(
153        const QString & noteContent, QString & plainText,
154        ErrorString & errorMessage);
155
156    static bool noteContentToListOfWords(
157        const QString & noteContent, QStringList & listOfWords,
158        ErrorString & errorMessage, QString * plainText = nullptr);
159
160    static QStringList plainTextToListOfWords(const QString & plainText);
161
162    static QString toDoCheckboxHtml(const bool checked, const quint64 idNumber);
163
164    static QString encryptedTextHtml(
165        const QString & encryptedText, const QString & hint,
166        const QString & cipher, const size_t keyLength,
167        const quint64 enCryptIndex);
168
169    static QString decryptedTextHtml(
170        const QString & decryptedText, const QString & encryptedText,
171        const QString & hint, const QString & cipher, const size_t keyLength,
172        const quint64 enDecryptedIndex);
173
174    static QString resourceHtml(
175        const Resource & resource, ErrorString & errorDescription);
176
177    static void escapeString(QString & string, const bool simplify = true);
178
179    enum class EnexExportTags
180    {
181        Yes = 0,
182        No
183    };
184
185    bool exportNotesToEnex(
186        const QVector<Note> & notes,
187        const QHash<QString, QString> & tagNamesByTagLocalUids,
188        const EnexExportTags exportTagsOption, QString & enex,
189        ErrorString & errorDescription, const QString & version = {}) const;
190
191    bool importEnex(
192        const QString & enex, QVector<Note> & notes,
193        QHash<QString, QStringList> & tagNamesByNoteLocalUid,
194        ErrorString & errorDescription) const;
195
196 private:
197     Q_DISABLE_COPY(ENMLConverter)
198
199 private:
200     ENMLConverterPrivate * const d_ptr;
201     Q_DECLARE_PRIVATE(ENMLConverter)
202 };
203

```

```

254 } // namespace quantier
255
256 #endif // LIB_QUENTIER_ENML_ENML_CONVERTER_H

```

## 6.3 HTMLCleaner.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_ENML_HTML_CLEANER_H
20 #define LIB_QUENTIER_ENML_HTML_CLEANER_H
21
22 #include <quantier/utility/Linkage.h>
23
24 #include <QString>
25
26 namespace quantier {
27
28 class QUENTIER_EXPORT HTMLCleaner
29 {
30 public:
31     HTMLCleaner();
32     virtual ~HTMLCleaner();
33
34     bool htmlToXml(
35         const QString & html, QString & output, QString & errorDescription);
36
37     bool htmlToXhtml(
38         const QString & html, QString & output, QString & errorDescription);
39
40     bool cleanupHtml(QString & html, QString & errorDescription);
41
42 private:
43     Q_DISABLE_COPY(HTMLCleaner)
44
45 private:
46     class Impl;
47     Impl * m_impl;
48 };
49
50 } // namespace quantier
51
52 #endif // LIB_QUENTIER_ENML_HTML_CLEANER_H

```

## 6.4 ApplicationSettingsInitializationException.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18

```

```

19 #ifndef LIB_QUENTIER_EXCEPTION_APPLICATION_SETTINGS_INITIALIZATION_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_APPLICATION_SETTINGS_INITIALIZATION_EXCEPTION_H
21
22 #include <quentier/exception/IQuentierException.h>
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT ApplicationSettingsInitializationException :
27     public IQuentierException
28 {
29 public:
30     explicit ApplicationSettingsInitializationException(
31         const ErrorString & message);
32
33 protected:
34     virtual const QString exceptionDisplayName() const override;
35 };
36
37 } // namespace quentier
38
39 #endif // LIB_QUENTIER_EXCEPTION_APPLICATION_SETTINGS_INITIALIZATION_EXCEPTION_H

```

## 6.5 DatabaseLockedException.h

```

1 /*
2  * Copyright 2016-2019 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_DATABASE_LOCKED_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_DATABASE_LOCKED_EXCEPTION_H
21
22 #include <quentier/exception/IQuentierException.h>
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT DatabaseLockedException : public IQuentierException
27 {
28 public:
29     explicit DatabaseLockedException(const ErrorString & message);
30
31 protected:
32     virtual const QString exceptionDisplayName() const override;
33 };
34
35 } // namespace quentier
36
37 #endif // LIB_QUENTIER_EXCEPTION_DATABASE_LOCKED_EXCEPTION_H

```

## 6.6 DatabaseLockFailedException.h

```

1 /*
2  * Copyright 2016-2019 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License

```

```

16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_DATABASE_LOCK_FAILED_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_DATABASE_LOCK_FAILED_EXCEPTION_H
21
22 #include <quentier/exception/IQuentierException.h>
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT DatabaseLockFailedException : public IQuentierException
27 {
28 public:
29     explicit DatabaseLockFailedException(const ErrorString & message);
30
31 protected:
32     virtual const QString exceptionDisplayName() const override;
33 };
34
35 } // namespace quentier
36
37 #endif // LIB_QUENTIER_EXCEPTION_DATABASE_LOCK_FAILED_EXCEPTION_H

```

## 6.7 DatabaseOpeningException.h

```

1 /*
2 * Copyright 2016-2019 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_DATABASE_OPENING_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_DATABASE_OPENING_EXCEPTION_H
21
22 #include <quentier/exception/IQuentierException.h>
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT DatabaseOpeningException : public IQuentierException
27 {
28 public:
29     explicit DatabaseOpeningException(const ErrorString & message);
30
31 protected:
32     virtual const QString exceptionDisplayName() const override;
33 };
34
35 } // namespace quentier
36
37 #endif // LIB_QUENTIER_EXCEPTION_DATABASE_OPENING_EXCEPTION_H

```

## 6.8 DatabaseRequestException.h

```

1 /*
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *

```



```

15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_DATABASE_SQL_ERROR_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_DATABASE_SQL_ERROR_EXCEPTION_H
21
22 #include <quentier/exception/IQuentierException.h>
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT DatabaseRequestException : public IQuentierException
27 {
28 public:
29     explicit DatabaseRequestException(const ErrorString & message);
30
31 protected:
32     virtual const QString exceptionDisplayName() const override;
33 };
34
35 } // namespace quentier
36
37 #endif // LIB_QUENTIER_EXCEPTION_DATABASE_SQL_ERROR_EXCEPTION_H

```

## 6.9 EmptyDataElementException.h

```

1 /*
2 * Copyright 2016-2019 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_EMPTY_DATA_ELEMENT_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_EMPTY_DATA_ELEMENT_EXCEPTION_H
21
22 #include <quentier/exception/IQuentierException.h>
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT EmptyDataElementException : public IQuentierException
27 {
28 public:
29     explicit EmptyDataElementException(const ErrorString & message);
30
31 protected:
32     virtual const QString exceptionDisplayName() const override;
33 };
34
35 } // namespace quentier
36
37 #endif // LIB_QUENTIER_EXCEPTION_EMPTY_DATA_ELEMENT_EXCEPTION_H

```

## 6.10 IQuentierException.h

```

1 /*
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.

```

```

14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_I_QUENTIER_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_I_QUENTIER_EXCEPTION_H
21
22 #include <exception>
23 #include <quentier/types/ErrorMessage.h>
24 #include <quentier/utility/Printable.h>
25
26 namespace quentier {
27
28 class QUENTIER_EXPORT IQuentierException :
29     public Printable,
30     public std::exception
31 {
32 public:
33     explicit IQuentierException(const ErrorMessage & message);
34
35     virtual ~IQuentierException() noexcept override;
36
37     QString localizedErrorMessage() const;
38     QString nonLocalizedErrorMessage() const;
39
40     virtual const char * what() const noexcept override;
41
42     virtual QTextStream & print(QTextStream & strm) const override;
43
44 protected:
45     IQuentierException(const IQuentierException & other);
46     IQuentierException & operator=(const IQuentierException & other);
47
48     virtual const QString exceptionDisplayName() const = 0;
49
50 private:
51     IQuentierException() = delete;
52
53     ErrorMessage m_message;
54     char * m_whatMessage;
55 };
56
57 } // namespace quentier
58
59 #endif // LIB_QUENTIER_EXCEPTION_I_QUENTIER_EXCEPTION_H

```

## 6.11 LocalStorageCacheManagerException.h

```

1 /*
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_LOCAL_STORAGE_CACHE_MANAGER_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_LOCAL_STORAGE_CACHE_MANAGER_EXCEPTION_H
21
22 #include <quentier/exception/IQuentierException.h>
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT LocalStorageCacheManagerException :
27     public IQuentierException
28 {
29 public:
30     explicit LocalStorageCacheManagerException(const ErrorMessage & message);
31
32 protected:
33     virtual const QString exceptionDisplayName() const override;
34 };

```

```

35
36 } // namespace quentier
37
38 #endif // LIB_QUENTIER_EXCEPTION_LOCAL_STORAGE_CACHE_MANAGER_EXCEPTION_H

```

## 6.12 LoggerInitializationException.h

```

1 /*
2  * Copyright 2016-2019 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_LOGGER_INITIALIZATION_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_LOGGER_INITIALIZATION_EXCEPTION_H
21
22 #include <quentier/exception/IQuentierException.h>
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT LoggerInitializationException : public IQuentierException
27 {
28 public:
29     explicit LoggerInitializationException(const ErrorString & message);
30
31 protected:
32     const QString exceptionDisplayName() const override;
33 };
34
35 } // namespace quentier
36
37 #endif // LIB_QUENTIER_EXCEPTION_LOGGER_INITIALIZATION_EXCEPTION_H

```

## 6.13 NoteEditorInitializationException.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_NOTE_EDITOR_INITIALIZATION_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_NOTE_EDITOR_INITIALIZATION_EXCEPTION_H
21
22 #include <quentier/exception/IQuentierException.h>
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT NoteEditorInitializationException :
27     public IQuentierException
28 {
29 public:
30     explicit NoteEditorInitializationException(const ErrorString & message);
31
32 protected:

```

```

33     virtual const QString exceptionDisplayName() const override;
34 };
35
36 } // namespace quantier
37
38 #endif // LIB_QUENTIER_EXCEPTION_NOTE_EDITOR_INITIALIZATION_EXCEPTION_H

```

## 6.14 NoteEditorPluginInitializationException.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_NOTE_EDITOR_PLUGIN_INITIALIZATION_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_NOTE_EDITOR_PLUGIN_INITIALIZATION_EXCEPTION_H
21
22 #include <quantier/exception/IQuantierException.h>
23
24 namespace quantier {
25
26 class QUENTIER_EXPORT NoteEditorPluginInitializationException :
27     public IQuantierException
28 {
29 public:
30     explicit NoteEditorPluginInitializationException(
31         const ErrorString & message);
32
33 protected:
34     virtual const QString exceptionDisplayName() const override;
35 };
36
37 } // namespace quantier
38
39 #endif // LIB_QUENTIER_EXCEPTION_NOTE_EDITOR_PLUGIN_INITIALIZATION_EXCEPTION_H

```

## 6.15 NullPtrException.h

```

1 /*
2  * Copyright 2016-2019 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_EXCEPTION_NULL_PTR_EXCEPTION_H
20 #define LIB_QUENTIER_EXCEPTION_NULL_PTR_EXCEPTION_H
21
22 #include <quantier/exception/IQuantierException.h>
23
24 namespace quantier {
25
26 class QUENTIER_EXPORT NullPtrException : public IQuantierException
27 {
28 public:

```

```

29     explicit NullPointerException(const ErrorString & message);
30
31 protected:
32     virtual const QString exceptionDisplayName() const override;
33 };
34
35 } // namespace quentier
36
37 #endif // LIB_QUENTIER_EXCEPTION_NULL_PTR_EXCEPTION_H

```

## 6.16 DefaultLocalStorageCacheExpiryChecker.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_LOCAL_STORAGE_DEFAULT_LOCAL_STORAGE_CACHE_EXPIRY_CHECKER_H
20 #define LIB_QUENTIER_LOCAL_STORAGE_DEFAULT_LOCAL_STORAGE_CACHE_EXPIRY_CHECKER_H
21
22 #include <quentier/local_storage/ILocalStorageCacheExpiryChecker.h>
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT DefaultLocalStorageCacheExpiryChecker :
27     public ILocalStorageCacheExpiryChecker
28 {
29 public:
30     DefaultLocalStorageCacheExpiryChecker(
31         const LocalStorageCacheManager & cacheManager);
32
33     virtual ~DefaultLocalStorageCacheExpiryChecker();
34
35     virtual DefaultLocalStorageCacheExpiryChecker * clone() const override;
36
37     virtual bool checkNotes() const override;
38
39     virtual bool checkResources() const override;
40
41     virtual bool checkNotebooks() const override;
42
43     virtual bool checkTags() const override;
44
45     virtual bool checkLinkedNotebooks() const override;
46
47     virtual bool checkSavedSearches() const override;
48
49     virtual QTextStream & print(QTextStream & strm) const override;
50
51 private:
52     Q_DISABLE_COPY(DefaultLocalStorageCacheExpiryChecker)
53 };
54
55 } // namespace quentier
56
57 #endif // LIB_QUENTIER_LOCAL_STORAGE_DEFAULT_LOCAL_STORAGE_CACHE_EXPIRY_CHECKER_H

```

## 6.17 ILocalStorageCacheExpiryChecker.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by

```

```

8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_LOCAL_STORAGE_I_LOCAL_STORAGE_CACHE_EXPIRY_CHECKER_H
20 #define LIB_QUENTIER_LOCAL_STORAGE_I_LOCAL_STORAGE_CACHE_EXPIRY_CHECKER_H
21
22 #include <quentier/utility/Printable.h>
23
24 namespace quentier {
25
26 QT_FORWARD_DECLARE_CLASS(LocalStorageCacheManager)
27
28
34 class QUENTIER_EXPORT ILocalStorageCacheExpiryChecker : public Printable
35 {
36 public:
37     virtual ~ILocalStorageCacheExpiryChecker();
38
39     virtual ILocalStorageCacheExpiryChecker * clone() const = 0;
40
41     virtual bool checkNotes() const = 0;
42
43     virtual bool checkResources() const = 0;
44
45     virtual bool checkNotebooks() const = 0;
46
47     virtual bool checkTags() const = 0;
48
49     virtual bool checkLinkedNotebooks() const = 0;
50
51     virtual bool checkSavedSearches() const = 0;
52
53     virtual QTextStream & print(QTextStream & strm) const = 0;
54
55 protected:
56     ILocalStorageCacheExpiryChecker(
57         const LocalStorageCacheManager & cacheManager);
58
59     const LocalStorageCacheManager & m_localStorageCacheManager;
60
61 private:
62     ILocalStorageCacheExpiryChecker() = delete;
63     Q_DISABLE_COPY(ILocalStorageCacheExpiryChecker)
64 };
65
66 } // namespace quentier
67
68 #endif // LIB_QUENTIER_LOCAL_STORAGE_I_LOCAL_STORAGE_CACHE_EXPIRY_CHECKER_H

```

## 6.18 ILocalStoragePatch.h

```

1 /*
2 * Copyright 2018-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_LOCAL_STORAGE_I_LOCAL_STORAGE_PATCH_H
20 #define LIB_QUENTIER_LOCAL_STORAGE_I_LOCAL_STORAGE_PATCH_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QObject>

```

```

25
26 namespace quentier {
27
28 QT_FORWARD_DECLARE_CLASS(ErrorString)
29 QT_FORWARD_DECLARE_CLASS(LocalStorageDatabaseUpgrader)
30
31
32 class QUENTIER_EXPORT ILocalStoragePatch : public QObject
33 {
34     Q_OBJECT
35 protected:
36     explicit ILocalStoragePatch(QObject * parent = nullptr);
37
38 public:
39     virtual ~ILocalStoragePatch();
40
41     virtual int fromVersion() const = 0;
42
43     virtual int toVersion() const = 0;
44
45     virtual QString patchShortDescription() const = 0;
46
47     virtual QString patchLongDescription() const = 0;
48
49     virtual bool backupLocalStorage(ErrorString & errorDescription) = 0;
50
51     virtual bool restoreLocalStorageFromBackup(
52         ErrorString & errorDescription) = 0;
53
54     virtual bool removeLocalStorageBackup(ErrorString & errorDescription) = 0;
55
56     virtual bool apply(ErrorString & errorDescription) = 0;
57
58     friend class LocalStorageDatabaseUpgrader;
59
60 Q_SIGNALS:
61     void progress(double progress);
62
63     void backupProgress(double progress);
64
65     void restoreBackupProgress(double progress);
66 };
67
68 } // namespace quentier
69
70 #endif // LIB_QUENTIER_LOCAL_STORAGE_I_LOCAL_STORAGE_PATCH_H

```

## 6.19 Lists.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_LOCAL_STORAGE_LISTS_H
20 #define LIB_QUENTIER_LOCAL_STORAGE_LISTS_H
21
22 #include <QVector>
23
24 namespace quentier {
25
26 QT_FORWARD_DECLARE_CLASS(LinkedNotebook)
27 QT_FORWARD_DECLARE_CLASS(Note)
28 QT_FORWARD_DECLARE_CLASS(Notebook)
29 QT_FORWARD_DECLARE_CLASS(Resource)
30 QT_FORWARD_DECLARE_CLASS(SavedSearch)
31 QT_FORWARD_DECLARE_CLASS(SharedNotebook)
32 QT_FORWARD_DECLARE_CLASS(Tag)
33 QT_FORWARD_DECLARE_CLASS(User)
34

```

```

35 using LinkedNotebookList = QList<LinkedNotebook>;
36 using NoteList = QList<Note>;
37 using NotebookList = QList<Notebook>;
38 using ResourceList = QList<Resource>;
39 using SavedSearchList = QList<SavedSearch>;
40 using SharedNotebookList = QList<SharedNotebook>;
41 using TagList = QList<Tag>;
42 using UserList = QList<User>;
43
44 } // namespace quentier
45
46 #endif // LIB_QUENTIER_LOCAL_STORAGE_LISTS_H

```

## 6.20 LocalStorageCacheManager.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_LOCAL_STORAGE_LOCAL_STORAGE_CACHE_MANAGER_H
20 #define LIB_QUENTIER_LOCAL_STORAGE_LOCAL_STORAGE_CACHE_MANAGER_H
21
22 #include <quentier/utility/Printable.h>
23
24 #include <memory>
25
26 namespace quentier {
27
28 QT_FORWARD_DECLARE_CLASS(LinkedNotebook)
29 QT_FORWARD_DECLARE_CLASS(Note)
30 QT_FORWARD_DECLARE_CLASS(Notebook)
31 QT_FORWARD_DECLARE_CLASS(Resource)
32 QT_FORWARD_DECLARE_CLASS(SavedSearch)
33 QT_FORWARD_DECLARE_CLASS(Tag)
34
35 QT_FORWARD_DECLARE_CLASS(ILocalStorageCacheExpiryChecker)
36
37 QT_FORWARD_DECLARE_CLASS(LocalStorageCacheManagerPrivate)
38 class QUENTIER_EXPORT LocalStorageCacheManager : public Printable
39 {
40 public:
41     LocalStorageCacheManager();
42     virtual ~LocalStorageCacheManager();
43
44     enum WhichUid
45     {
46         LocalUid,
47         Guid
48     };
49
50     void clear();
51     bool empty() const;
52
53     // Notes cache
54     size_t numCachedNotes() const;
55     void cacheNote(const Note & note);
56     void expungeNote(const Note & note);
57
58     const Note * findNote(const QString & uid, const WhichUid whichUid) const;
59
60     void clearAllNotes();
61
62     // Resources cache
63     size_t numCachedResources() const;
64     void cacheResource(const Resource & resource);
65     void expungeResource(const Resource & resource);
66
67     const Resource * findResource(
68         const QString & id, const WhichUid whichUid) const;

```



```

69
70     void clearAllResources();
71
72     // Notebooks cache
73     size_t numCachedNotebooks() const;
74     void cacheNotebook(const Notebook & notebook);
75     void expungeNotebook(const Notebook & notebook);
76
77     const Notebook * findNotebook(
78         const QString & uid, const WhichUid whichUid) const;
79
80     const Notebook * findNotebookByName(const QString & name) const;
81     void clearAllNotebooks();
82
83     // Tags cache
84     size_t numCachedTags() const;
85     void cacheTag(const Tag & tag);
86     void expungeTag(const Tag & tag);
87     const Tag * findTag(const QString & uid, const WhichUid whichUid) const;
88     const Tag * findTagByName(const QString & name) const;
89     void clearAllTags();
90
91     // Linked notebooks cache
92     size_t numCachedLinkedNotebooks() const;
93     void cacheLinkedNotebook(const LinkedNotebook & linkedNotebook);
94     void expungeLinkedNotebook(const LinkedNotebook & linkedNotebook);
95     const LinkedNotebook * findLinkedNotebook(const QString & guid) const;
96     void clearAllLinkedNotebooks();
97
98     // Saved searches cache
99     size_t numCachedSavedSearches() const;
100    void cacheSavedSearch(const SavedSearch & savedSearch);
101    void expungeSavedSearch(const SavedSearch & savedSearch);
102
103    const SavedSearch * findSavedSearch(
104        const QString & uid, const WhichUid whichUid) const;
105
106    const SavedSearch * findSavedSearchByName(const QString & name) const;
107    void clearAllSavedSearches();
108
109    void installCacheExpiryFunction(
110        const ILocalStorageCacheExpiryChecker & checker);
111
112    virtual QTextStream & print(QTextStream & strm) const override;
113
114 private:
115     Q_DISABLE_COPY(LocalStorageCacheManager)
116
117     LocalStorageCacheManagerPrivate * const d_ptr;
118     Q_DECLARE_PRIVATE(LocalStorageCacheManager)
119 };
120
121 } // namespace quantier
122
123 #endif // LIB_QUENTIER_LOCAL_STORAGE_LOCAL_STORAGE_CACHE_MANAGER_H

```

## 6.21 LocalStorageManager.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_LOCAL_STORAGE_LOCAL_STORAGE_MANAGER_H
20 #define LIB_QUENTIER_LOCAL_STORAGE_LOCAL_STORAGE_MANAGER_H
21
22 #include <quantier/local_storage/Lists.h>
23 #include <quantier/local_storage/NoteSearchQuery.h>
24 #include <quantier/types/Account.h>
25 #include <quantier/types/ErrorMessage.h>

```

```

26 #include <quentier/utility/Linkage.h>
27
28 #include <QHash>
29 #include <QString>
30 #include <QVector>
31
32 #include <cstdint>
33 #include <memory>
34 #include <utility>
35
36 namespace qevercloud {
37
38 QT_FORWARD_DECLARE_STRUCT(Accounting)
39 QT_FORWARD_DECLARE_STRUCT(BusinessUserInfo)
40 QT_FORWARD_DECLARE_STRUCT(NoteAttributes)
41 QT_FORWARD_DECLARE_STRUCT(NotebookRestrictions)
42 QT_FORWARD_DECLARE_STRUCT(ResourceAttributes)
43 QT_FORWARD_DECLARE_STRUCT(PremiumInfo)
44 QT_FORWARD_DECLARE_STRUCT(SharedNotebook)
45 QT_FORWARD_DECLARE_STRUCT(UserAttributes)
46
47 } // namespace qevercloud
48
49 namespace quentier {
50
51 QT_FORWARD_DECLARE_CLASS(ILocalStoragePatch)
52 QT_FORWARD_DECLARE_CLASS(LocalStorageManagerPrivate)
53
54 class QUENTIER_EXPORT LocalStorageManager : public QObject
55 {
56     Q_OBJECT
57 public:
58     enum class StartupOption
59     {
60         ClearDatabase = 1,
61         OverrideLock = 2
62     };
63     Q_DECLARE_FLAGS(StartupOptions, StartupOption)
64
65     friend QUENTIER_EXPORT QTextStream & operator<<(
66         QTextStream & strm, const StartupOption option);
67
68     friend QUENTIER_EXPORT QDebug & operator<<(
69         QDebug & dbg, const StartupOption option);
70
71     friend QUENTIER_EXPORT QTextStream & operator<<(
72         QTextStream & strm, const StartupOptions options);
73
74     friend QUENTIER_EXPORT QDebug & operator<<(
75         QDebug & dbg, const StartupOptions options);
76
77     explicit LocalStorageManager(
78         const Account & account,
79         #if QT_VERSION >= QT_VERSION_CHECK(5, 15, 0)
80         const StartupOptions options = {},
81         #else
82         const StartupOptions options = 0,
83         #endif
84         QObject * parent = nullptr);
85
86     virtual ~LocalStorageManager() override;
87
88 Q_SIGNALS:
89     void upgradeProgress(double progress);
90
91 public:
92     enum class ListObjectsOption
93     {
94         ListAll = 0,
95         ListDirty = 1,
96         ListNonDirty = 2,
97         ListElementsWithoutGuid = 4,
98         ListElementsWithGuid = 8,
99         ListLocal = 16,
100        ListNonLocal = 32,
101        ListFavoritedElements = 64,
102        ListNonFavoritedElements = 128
103    };
104    Q_DECLARE_FLAGS(ListObjectsOptions, ListObjectsOption)
105
106    friend QUENTIER_EXPORT QTextStream & operator<<(
107        QTextStream & strm, const ListObjectsOption option);
108
109    friend QUENTIER_EXPORT QDebug & operator<<(
110        QDebug & dbg, const ListObjectsOption option);
111
112    friend QUENTIER_EXPORT QTextStream & operator<<(

```

```

164     QTextStream & strm, const ListObjectsOptions options);
165
166     friend QUINTIER_EXPORT QDebug & operator<< (
167         QDebug & dbg, const ListObjectsOptions options);
168
169     void switchUser (
170         const Account & account,
171         #if QT_VERSION >= QT_VERSION_CHECK(5, 15, 0)
172         const StartupOptions options = {});
173     #else
174         const StartupOptions options = 0);
175     #endif
176
177     bool isLocalStorageVersionTooHigh(ErrorString & errorDescription);
178
179     bool localStorageRequiresUpgrade(ErrorString & errorDescription);
180
181     QVector<std::shared_ptr<ILocalStoragePatch>> requiredLocalStoragePatches();
182
183     qint32 localStorageVersion(ErrorString & errorDescription);
184
185     qint32 highestSupportedLocalStorageVersion() const;
186
187     int userCount(ErrorString & errorDescription) const;
188
189     bool addUser(const User & user, ErrorString & errorDescription);
190
191     bool updateUser(const User & user, ErrorString & errorDescription);
192
193     bool findUser(User & user, ErrorString & errorDescription) const;
194
195     bool deleteUser(const User & user, ErrorString & errorDescription);
196
197     bool expungeUser(const User & user, ErrorString & errorDescription);
198
199     int notebookCount(ErrorString & errorDescription) const;
200
201     bool addNotebook(Notebook & notebook, ErrorString & errorDescription);
202
203     bool updateNotebook(Notebook & notebook, ErrorString & errorDescription);
204
205     bool findNotebook (
206         Notebook & notebook, ErrorString & errorDescription) const;
207
208     bool findDefaultNotebook (
209         Notebook & notebook, ErrorString & errorDescription) const;
210
211     bool findLastUsedNotebook (
212         Notebook & notebook, ErrorString & errorDescription) const;
213
214     bool findDefaultOrLastUsedNotebook (
215         Notebook & notebook, ErrorString & errorDescription) const;
216
217     enum class OrderDirection
218     {
219         Ascending = 0,
220         Descending
221     };
222
223     friend QUINTIER_EXPORT QTextStream & operator<< (
224         QTextStream & strm, const OrderDirection orderDirection);
225
226     friend QUINTIER_EXPORT QDebug & operator<< (
227         QDebug & dbg, const OrderDirection orderDirection);
228
229     enum class ListNotebooksOrder
230     {
231         ByUpdateSequenceNumber = 0,
232         ByNotebookName,
233         ByCreationTimestamp,
234         ByModificationTimestamp,
235         NoOrder
236     };
237
238     friend QUINTIER_EXPORT QTextStream & operator<< (
239         QTextStream & strm, const ListNotebooksOrder order);
240
241     friend QUINTIER_EXPORT QDebug & operator<< (
242         QDebug & dbg, const ListNotebooksOrder order);
243
244     QList<Notebook> listAllNotebooks (
245         ErrorString & errorDescription, const size_t limit = 0,
246         const size_t offset = 0,
247         const ListNotebooksOrder order = ListNotebooksOrder::NoOrder,
248         const OrderDirection orderDirection = OrderDirection::Ascending,
249         const QString & linkedNotebookGuid = QString()) const;
250

```

```

596 QList<Notebook> listNotebooks(
597     const ListObjectsOptions flag, ErrorString & errorDescription,
598     const size_t limit = 0, const size_t offset = 0,
599     const ListNotebooksOrder order = ListNotebooksOrder::NoOrder,
600     const OrderDirection orderDirection = OrderDirection::Ascending,
601     const QString & linkedNotebookGuid = QString()) const;
602
603 QList<SharedNotebook> listAllSharedNotebooks(
604     ErrorString & errorDescription) const;
605
606 QList<SharedNotebook> listSharedNotebooksPerNotebookGuid(
607     const QString & notebookGuid, ErrorString & errorDescription) const;
608
609 bool expungeNotebook(Notebook & notebook, ErrorString & errorDescription);
610
611 int linkedNotebookCount(ErrorString & errorDescription) const;
612
613 bool addLinkedNotebook(
614     const LinkedNotebook & linkedNotebook, ErrorString & errorDescription);
615
616 bool updateLinkedNotebook(
617     const LinkedNotebook & linkedNotebook, ErrorString & errorDescription);
618
619 bool findLinkedNotebook(
620     LinkedNotebook & linkedNotebook, ErrorString & errorDescription) const;
621
622 enum class ListLinkedNotebooksOrder
623 {
624     ByUpdateSequenceNumber = 0,
625     ByShareName,
626     ByUsername,
627     NoOrder
628 };
629
630 friend QUENTIER_EXPORT QTextStream & operator<<(
631     QTextStream & strm, const ListLinkedNotebooksOrder order);
632
633 friend QUENTIER_EXPORT QDebug & operator<<(
634     QDebug & strm, const ListLinkedNotebooksOrder order);
635
636 QList<LinkedNotebook> listAllLinkedNotebooks(
637     ErrorString & errorDescription, const size_t limit = 0,
638     const size_t offset = 0,
639     const ListLinkedNotebooksOrder order =
640         ListLinkedNotebooksOrder::NoOrder,
641     const OrderDirection orderDirection = OrderDirection::Ascending) const;
642
643 QList<LinkedNotebook> listLinkedNotebooks(
644     const ListObjectsOptions flag, ErrorString & errorDescription,
645     const size_t limit = 0, const size_t offset = 0,
646     const ListLinkedNotebooksOrder order =
647         ListLinkedNotebooksOrder::NoOrder,
648     const OrderDirection orderDirection = OrderDirection::Ascending) const;
649
650 bool expungeLinkedNotebook(
651     const LinkedNotebook & linkedNotebook, ErrorString & errorDescription);
652
653 enum class NoteCountOption
654 {
655     IncludeNonDeletedNotes = 1,
656     IncludeDeletedNotes = 2
657 };
658 Q_DECLARE_FLAGS(NoteCountOptions, NoteCountOption)
659
660 friend QUENTIER_EXPORT QTextStream & operator<<(
661     QTextStream & strm, const NoteCountOption option);
662
663 friend QUENTIER_EXPORT QDebug & operator<<(
664     QDebug & dbg, const NoteCountOption option);
665
666 friend QUENTIER_EXPORT QTextStream & operator<<(
667     QTextStream & strm, const NoteCountOptions options);
668
669 friend QUENTIER_EXPORT QDebug & operator<<(
670     QDebug & strm, const NoteCountOptions options);
671
672 int noteCount(
673     ErrorString & errorDescription,
674     const NoteCountOptions options =
675         NoteCountOption::IncludeNonDeletedNotes) const;
676
677 int noteCountPerNotebook(
678     const Notebook & notebook, ErrorString & errorDescription,
679     const NoteCountOptions options =
680         NoteCountOption::IncludeNonDeletedNotes) const;
681
682 int noteCountPerTag(

```

```

902     const Tag & tag, ErrorString & errorDescription,
903     const NoteCountOptions options =
904         NoteCountOption::IncludeNonDeletedNotes) const;
905
906 bool noteCountsPerAllTags(
907     QHash<QString, int> & noteCountsPerTagLocalUid,
908     ErrorString & errorDescription,
909     const NoteCountOptions options =
910         NoteCountOption::IncludeNonDeletedNotes) const;
911
912 int noteCountPerNotebooksAndTags(
913     const QStringList & notebookLocalUids, const QStringList & tagLocalUids,
914     ErrorString & errorDescription,
915     const NoteCountOptions options =
916         NoteCountOption::IncludeNonDeletedNotes) const;
917
918 bool addNote(Note & note, ErrorString & errorDescription);
919
920 enum class UpdateNoteOption
921 {
922     UpdateResourceMetadata = 1,
923     UpdateResourceBinaryData = 2,
924     UpdateTags = 4
925 };
926 Q_DECLARE_FLAGS(UpdateNoteOptions, UpdateNoteOption)
927
928 friend QUINTIER_EXPORT QTextStream & operator<<(
929     QTextStream & strm, const UpdateNoteOption option);
930
931 friend QUINTIER_EXPORT QDebug & operator<<(
932     QDebug & strm, const UpdateNoteOption option);
933
934 friend QUINTIER_EXPORT QTextStream & operator<<(
935     QTextStream & strm, const UpdateNoteOptions options);
936
937 friend QUINTIER_EXPORT QDebug & operator<<(
938     QDebug & strm, const UpdateNoteOptions options);
939
940 bool updateNote(
941     Note & note, const UpdateNoteOptions options,
942     ErrorString & errorDescription);
943
944 enum class GetNoteOption
945 {
946     WithResourceMetadata = 1,
947     WithResourceBinaryData = 2
948 };
949 Q_DECLARE_FLAGS(GetNoteOptions, GetNoteOption)
950
951 friend QUINTIER_EXPORT QTextStream & operator<<(
952     QTextStream & strm, const GetNoteOption option);
953
954 friend QUINTIER_EXPORT QDebug & operator<<(
955     QDebug & dbg, const GetNoteOption option);
956
957 friend QUINTIER_EXPORT QTextStream & operator<<(
958     QTextStream & strm, const GetNoteOptions options);
959
960 friend QUINTIER_EXPORT QDebug & operator<<(
961     QDebug & strm, const GetNoteOptions options);
962
963 bool findNote(
964     Note & note, const GetNoteOptions options,
965     ErrorString & errorDescription) const;
966
967 enum class ListNotesOrder
968 {
969     ByUpdateSequenceNumber = 0,
970     ByTitle,
971     ByCreationTimestamp,
972     ByModificationTimestamp,
973     ByDeletionTimestamp,
974     ByAuthor,
975     BySource,
976     BySourceApplication,
977     ByReminderTime,
978     ByPlaceName,
979     NoOrder
980 };
981
982 friend QUINTIER_EXPORT QTextStream & operator<<(
983     QTextStream & strm, const ListNotesOrder order);
984
985 friend QUINTIER_EXPORT QDebug & operator<<(
986     QDebug & strm, const ListNotesOrder order);
987
988 QList<Note> listNotesPerNotebook(

```

```

1168     const Notebook & notebook, const GetNoteOptions options,
1169     ErrorString & errorDescription,
1170     const ListObjectsOptions & flag = ListObjectsOption::ListAll,
1171     const size_t limit = 0, const size_t offset = 0,
1172     const ListNotesOrder & order = ListNotesOrder::NoOrder,
1173     const OrderDirection & orderDirection =
1174         OrderDirection::Ascending) const;
1175
1203 QList<Note> listNotesPerTag(
1204     const Tag & tag, const GetNoteOptions options,
1205     ErrorString & errorDescription,
1206     const ListObjectsOptions & flag = ListObjectsOption::ListAll,
1207     const size_t limit = 0, const size_t offset = 0,
1208     const ListNotesOrder & order = ListNotesOrder::NoOrder,
1209     const OrderDirection & orderDirection =
1210         OrderDirection::Ascending) const;
1211
1242 QList<Note> listNotesPerNotebooksAndTags(
1243     const QStringList & notebookLocalUids, const QStringList & tagLocalUids,
1244     const GetNoteOptions options, ErrorString & errorDescription,
1245     const ListObjectsOptions & flag = ListObjectsOption::ListAll,
1246     const size_t limit = 0, const size_t offset = 0,
1247     const ListNotesOrder & order = ListNotesOrder::NoOrder,
1248     const OrderDirection & orderDirection =
1249         OrderDirection::Ascending) const;
1250
1282 QList<Note> listNotesByLocalUids(
1283     const QStringList & noteLocalUids, const GetNoteOptions options,
1284     ErrorString & errorDescription,
1285     const ListObjectsOptions & flag = ListObjectsOption::ListAll,
1286     const size_t limit = 0, const size_t offset = 0,
1287     const ListNotesOrder & order = ListNotesOrder::NoOrder,
1288     const OrderDirection & orderDirection =
1289         OrderDirection::Ascending) const;
1290
1325 QList<Note> listNotes(
1326     const ListObjectsOptions flag, const GetNoteOptions options,
1327     ErrorString & errorDescription, const size_t limit = 0,
1328     const size_t offset = 0,
1329     const ListNotesOrder order = ListNotesOrder::NoOrder,
1330     const OrderDirection orderDirection = OrderDirection::Ascending,
1331     const QString & linkedNotebookGuid = QString()) const;
1332
1344 QStringList findNoteLocalUidsWithSearchQuery(
1345     const NoteSearchQuery & noteSearchQuery,
1346     ErrorString & errorDescription) const;
1347
1363 NoteList findNotesWithSearchQuery(
1364     const NoteSearchQuery & noteSearchQuery, const GetNoteOptions options,
1365     ErrorString & errorDescription) const;
1366
1384 bool expungeNote(Note & note, ErrorString & errorDescription);
1385
1395 int tagCount(ErrorString & errorDescription) const;
1396
1410 bool addTag(Tag & tag, ErrorString & errorDescription);
1411
1429 bool updateTag(Tag & tag, ErrorString & errorDescription);
1430
1455 bool findTag(Tag & tag, ErrorString & errorDescription) const;
1456
1461 enum class ListTagsOrder
1462 {
1463     ByUpdateSequenceNumber,
1464     ByName,
1465     NoOrder
1466 };
1467
1468 friend QUENTIER_EXPORT QTextStream & operator<<(
1469     QTextStream & strm, const ListTagsOrder order);
1470
1471 friend QUENTIER_EXPORT QDebug & operator<<(
1472     QDebug & strm, const ListTagsOrder order);
1473
1502 QList<Tag> listAllTagsPerNote(
1503     const Note & note, ErrorString & errorDescription,
1504     const ListObjectsOptions & flag = ListObjectsOption::ListAll,
1505     const size_t limit = 0, const size_t offset = 0,
1506     const ListTagsOrder & order = ListTagsOrder::NoOrder,
1507     const OrderDirection & orderDirection =
1508         OrderDirection::Ascending) const;
1509
1540 QList<Tag> listAllTags(
1541     ErrorString & errorDescription, const size_t limit = 0,
1542     const size_t offset = 0,
1543     const ListTagsOrder order = ListTagsOrder::NoOrder,
1544     const OrderDirection orderDirection = OrderDirection::Ascending,

```

```

1545         const QString & linkedNotebookGuid = QString()) const;
1546
1579 QList<Tag> listTags(
1580     const ListObjectsOptions flag, ErrorString & errorDescription,
1581     const size_t limit = 0, const size_t offset = 0,
1582     const ListTagsOrder & order = ListTagsOrder::NoOrder,
1583     const OrderDirection orderDirection = OrderDirection::Ascending,
1584     const QString & linkedNotebookGuid = QString()) const;
1585
1624 QList<std::pair<Tag, QStringList>> listTagsWithNoteLocalUids(
1625     const ListObjectsOptions flag, ErrorString & errorDescription,
1626     const size_t limit = 0, const size_t offset = 0,
1627     const ListTagsOrder & order = ListTagsOrder::NoOrder,
1628     const OrderDirection orderDirection = OrderDirection::Ascending,
1629     const QString & linkedNotebookGuid = QString()) const;
1630
1655 bool expungeTag(
1656     Tag & tag, QStringList & expungedChildTagLocalUids,
1657     ErrorString & errorDescription);
1658
1669 bool expungeNotelessTagsFromLinkedNotebooks(ErrorString & errorDescription);
1670
1682 int enResourceCount(ErrorString & errorDescription) const;
1683
1700 bool addEnResource(Resource & resource, ErrorString & errorDescription);
1701
1722 bool updateEnResource(Resource & resource, ErrorString & errorDescription);
1723
1734 enum class GetResourceOption
1735 {
1740     WithBinaryData = 1
1741 };
1742 Q_DECLARE_FLAGS(GetResourceOptions, GetResourceOption)
1743
1744 friend QUINTIER_EXPORT QTextStream & operator<<(
1745     QTextStream & strm, const GetResourceOption option);
1746
1747 friend QUINTIER_EXPORT QDebug & operator<<(
1748     QDebug & strm, const GetResourceOption option);
1749
1750 friend QUINTIER_EXPORT QTextStream & operator<<(
1751     QTextStream & strm, const GetResourceOptions options);
1752
1753 friend QUINTIER_EXPORT QDebug & operator<<(
1754     QDebug & strm, const GetResourceOptions options);
1755
1774 bool findEnResource(
1775     Resource & resource, const GetResourceOptions options,
1776     ErrorString & errorDescription) const;
1777
1792 bool expungeEnResource(Resource & resource, ErrorString & errorDescription);
1793
1803 int savedSearchCount(ErrorString & errorDescription) const;
1804
1821 bool addSavedSearch(SavedSearch & search, ErrorString & errorDescription);
1822
1841 bool updateSavedSearch(
1842     SavedSearch & search, ErrorString & errorDescription);
1843
1861 bool findSavedSearch(
1862     SavedSearch & search, ErrorString & errorDescription) const;
1863
1868 enum class ListSavedSearchesOrder
1869 {
1870     ByUpdateSequenceNumber = 0,
1871     ByName,
1872     ByFormat,
1873     NoOrder
1874 };
1875
1876 friend QUINTIER_EXPORT QTextStream & operator<<(
1877     QTextStream & strm, const ListSavedSearchesOrder order);
1878
1879 friend QUINTIER_EXPORT QDebug & operator<<(
1880     QDebug & strm, const ListSavedSearchesOrder order);
1881
1904 QList<SavedSearch> listAllSavedSearches(
1905     ErrorString & errorDescription, const size_t limit = 0,
1906     const size_t offset = 0,
1907     const ListSavedSearchesOrder order = ListSavedSearchesOrder::NoOrder,
1908     const OrderDirection orderDirection = OrderDirection::Ascending) const;
1909
1939 QList<SavedSearch> listSavedSearches(
1940     const ListObjectsOptions flag, ErrorString & errorDescription,
1941     const size_t limit = 0, const size_t offset = 0,
1942     const ListSavedSearchesOrder order = ListSavedSearchesOrder::NoOrder,
1943     const OrderDirection orderDirection = OrderDirection::Ascending) const;

```

```

1944
1958     bool expungeSavedSearch(
1959         SavedSearch & search, QString & errorDescription);
1960
1978     qint32 accountHighUsn(
1979         const QString & linkedNotebookGuid, QString & errorDescription);
1980
1981 private:
1982     Q_DISABLE_COPY(LocalStorageManager)
1983
1984     LocalStorageManagerPrivate * const d_ptr;
1985     Q_DECLARE_PRIVATE(LocalStorageManager)
1986 };
1987
1988 Q_DECLARE_OPERATORS_FOR_FLAGS(LocalStorageManager::GetNoteOptions)
1989 Q_DECLARE_OPERATORS_FOR_FLAGS(LocalStorageManager::ListObjectsOptions)
1990 Q_DECLARE_OPERATORS_FOR_FLAGS(LocalStorageManager::StartupOptions)
1991 Q_DECLARE_OPERATORS_FOR_FLAGS(LocalStorageManager::UpdateNoteOptions)
1992
1993 } // namespace quantier
1994
1995 #endif // LIB_QUENTIER_LOCAL_STORAGE_LOCAL_STORAGE_MANAGER_H

```

## 6.22 LocalStorageManagerAsync.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_LOCAL_STORAGE_LOCAL_STORAGE_MANAGER_ASYNC_H
20 #define LIB_QUENTIER_LOCAL_STORAGE_LOCAL_STORAGE_MANAGER_ASYNC_H
21
22 #include <quantier/local_storage/ILocalStorageCacheExpiryChecker.h>
23 #include <quantier/local_storage/LocalStorageCacheManager.h>
24 #include <quantier/local_storage/LocalStorageManager.h>
25 #include <quantier/types/ErrorMessage.h>
26 #include <quantier/types/LinkedNotebook.h>
27 #include <quantier/types/Note.h>
28 #include <quantier/types/Notebook.h>
29 #include <quantier/types/Resource.h>
30 #include <quantier/types/SavedSearch.h>
31 #include <quantier/types/SharedNotebook.h>
32 #include <quantier/types/Tag.h>
33 #include <quantier/types/User.h>
34
35 #include <QObject>
36
37 #include <memory>
38
39 namespace quantier {
40
41 QT_FORWARD_DECLARE_CLASS(LocalStorageManagerAsyncPrivate)
42
43 class QUENTIER_EXPORT LocalStorageManagerAsync : public QObject
44 {
45     Q_OBJECT
46 public:
47     explicit LocalStorageManagerAsync(
48         const Account & account,
49         #if QT_VERSION >= QT_VERSION_CHECK(5, 15, 0)
50         LocalStorageManager::StartupOptions options = {},
51     #else
52         LocalStorageManager::StartupOptions options = 0,
53     #endif
54         QObject * parent = nullptr);
55
56     virtual ~LocalStorageManagerAsync();
57
58     void setUseCache(const bool useCache);

```



```

59
60     const LocalStorageCacheManager * localStorageCacheManager() const;
61
62     bool installCacheExpiryFunction(
63         const ILocalStorageCacheExpiryChecker & checker);
64
65     const LocalStorageManager * localStorageManager() const;
66     LocalStorageManager * localStorageManager();
67
68     Q_SIGNALS:
69         // Sent when the initialization is complete
70         void initialized();
71
72         // User-related signals:
73         void getUserCountComplete(int userCount, QUuid requestId);
74         void getUserCountFailed(ErrorString errorDescription, QUuid requestId);
75         void switchUserComplete(Account account, QUuid requestId);
76
77         void switchUserFailed(
78             Account account, ErrorString errorDescription, QUuid requestId);
79
80         void addUserComplete(User user, QUuid requestId);
81
82         void addUserFailed(
83             User user, ErrorString errorDescription, QUuid requestId);
84
85         void updateUserComplete(User user, QUuid requestId);
86
87         void updateUserFailed(
88             User user, ErrorString errorDescription, QUuid requestId);
89
90         void findUserComplete(User foundUser, QUuid requestId);
91
92         void findUserFailed(
93             User user, ErrorString errorDescription, QUuid requestId);
94
95         void deleteUserComplete(User user, QUuid requestId);
96
97         void deleteUserFailed(
98             User user, ErrorString errorDescription, QUuid requestId);
99
100        void expungeUserComplete(User user, QUuid requestId);
101
102        void expungeUserFailed(
103            User user, ErrorString errorDescription, QUuid requestId);
104
105        // Notebook-related signals:
106        void getNotebookCountComplete(int notebookCount, QUuid requestId);
107        void getNotebookCountFailed(ErrorString errorDescription, QUuid requestId);
108        void addNotebookComplete(Notebook notebook, QUuid requestId);
109
110        void addNotebookFailed(
111            Notebook notebook, ErrorString errorDescription, QUuid requestId);
112
113        void updateNotebookComplete(Notebook notebook, QUuid requestId);
114
115        void updateNotebookFailed(
116            Notebook notebook, ErrorString errorDescription, QUuid requestId);
117
118        void findNotebookComplete(Notebook foundNotebook, QUuid requestId);
119
120        void findNotebookFailed(
121            Notebook notebook, ErrorString errorDescription, QUuid requestId);
122
123        void findDefaultNotebookComplete(Notebook foundNotebook, QUuid requestId);
124
125        void findDefaultNotebookFailed(
126            Notebook notebook, ErrorString errorDescription, QUuid requestId);
127
128        void findLastUsedNotebookComplete(Notebook foundNotebook, QUuid requestId);
129
130        void findLastUsedNotebookFailed(
131            Notebook notebook, ErrorString errorDescription, QUuid requestId);
132
133        void findDefaultOrLastUsedNotebookComplete(
134            Notebook foundNotebook, QUuid requestId);
135
136        void findDefaultOrLastUsedNotebookFailed(
137            Notebook notebook, ErrorString errorDescription, QUuid requestId);
138
139        void listAllNotebooksComplete(
140            size_t limit, size_t offset,
141            LocalStorageManager::ListNotebooksOrder order,
142            LocalStorageManager::OrderDirection orderDirection,
143            QString linkedNotebookGuid, QList<Notebook> foundNotebooks,
144            QUuid requestId);
145

```

```

146 void listAllNotebooksFailed(
147     size_t limit, size_t offset,
148     LocalStorageManager::ListNotebooksOrder order,
149     LocalStorageManager::OrderDirection orderDirection,
150     QString linkedNotebookGuid, ErrorString errorDescription,
151     QUuid requestId);
152
153 void listNotebooksComplete(
154     LocalStorageManager::ListObjectsOptions flag, size_t limit,
155     size_t offset, LocalStorageManager::ListNotebooksOrder order,
156     LocalStorageManager::OrderDirection orderDirection,
157     QString linkedNotebookGuid, QList<Notebook> foundNotebooks,
158     QUuid requestId);
159
160 void listNotebooksFailed(
161     LocalStorageManager::ListObjectsOptions flag, size_t limit,
162     size_t offset, LocalStorageManager::ListNotebooksOrder order,
163     LocalStorageManager::OrderDirection orderDirection,
164     QString linkedNotebookGuid, ErrorString errorDescription,
165     QUuid requestId);
166
167 void listAllSharedNotebooksComplete(
168     QList<SharedNotebook> foundSharedNotebooks, QUuid requestId);
169
170 void listAllSharedNotebooksFailed(
171     ErrorString errorDescription, QUuid requestId);
172
173 void listSharedNotebooksPerNotebookGuidComplete(
174     QString notebookGuid, QList<SharedNotebook> foundSharedNotebooks,
175     QUuid requestId);
176
177 void listSharedNotebooksPerNotebookGuidFailed(
178     QString notebookGuid, ErrorString errorDescription, QUuid requestId);
179
180 void expungeNotebookComplete(Notebook notebook, QUuid requestId);
181
182 void expungeNotebookFailed(
183     Notebook notebook, ErrorString errorDescription, QUuid requestId);
184
185 // Linked notebook-related signals:
186 void getLinkedNotebookCountComplete(
187     int linkedNotebookCount, QUuid requestId);
188
189 void getLinkedNotebookCountFailed(
190     ErrorString errorDescription, QUuid requestId);
191
192 void addLinkedNotebookComplete(
193     LinkedNotebook linkedNotebook, QUuid requestId);
194
195 void addLinkedNotebookFailed(
196     LinkedNotebook linkedNotebook, ErrorString errorDescription,
197     QUuid requestId);
198
199 void updateLinkedNotebookComplete(
200     LinkedNotebook linkedNotebook, QUuid requestId);
201
202 void updateLinkedNotebookFailed(
203     LinkedNotebook linkedNotebook, ErrorString errorDescription,
204     QUuid requestId);
205
206 void findLinkedNotebookComplete(
207     LinkedNotebook foundLinkedNotebook, QUuid requestId);
208
209 void findLinkedNotebookFailed(
210     LinkedNotebook linkedNotebook, ErrorString errorDescription,
211     QUuid requestId);
212
213 void listAllLinkedNotebooksComplete(
214     size_t limit, size_t offset,
215     LocalStorageManager::ListLinkedNotebooksOrder order,
216     LocalStorageManager::OrderDirection orderDirection,
217     QList<LinkedNotebook> foundLinkedNotebooks, QUuid requestId);
218
219 void listAllLinkedNotebooksFailed(
220     size_t limit, size_t offset,
221     LocalStorageManager::ListLinkedNotebooksOrder order,
222     LocalStorageManager::OrderDirection orderDirection,
223     ErrorString errorDescription, QUuid requestId);
224
225 void listLinkedNotebooksComplete(
226     LocalStorageManager::ListObjectsOptions flag, size_t limit,
227     size_t offset, LocalStorageManager::ListLinkedNotebooksOrder order,
228     LocalStorageManager::OrderDirection orderDirection,
229     QList<LinkedNotebook> foundLinkedNotebooks, QUuid requestId);
230
231 void listLinkedNotebooksFailed(
232     LocalStorageManager::ListObjectsOptions flag, size_t limit,

```

```

233         size_t offset, LocalStorageManager::ListLinkedNotebooksOrder order,
234         LocalStorageManager::OrderDirection orderDirection,
235         ErrorString errorDescription, QUuid requestId);
236
237     void expungeLinkedNotebookComplete(
238         LinkedNotebook linkedNotebook, QUuid requestId);
239
240     void expungeLinkedNotebookFailed(
241         LinkedNotebook linkedNotebook, ErrorString errorDescription,
242         QUuid requestId);
243
244     // Note-related signals:
245     void getNoteCountComplete(
246         int noteCount, LocalStorageManager::NoteCountOptions options,
247         QUuid requestId);
248
249     void getNoteCountFailed(
250         ErrorString errorDescription,
251         LocalStorageManager::NoteCountOptions options, QUuid requestId);
252
253     void getNoteCountPerNotebookComplete(
254         int noteCount, Notebook notebook,
255         LocalStorageManager::NoteCountOptions options, QUuid requestId);
256
257     void getNoteCountPerNotebookFailed(
258         ErrorString errorDescription, Notebook notebook,
259         LocalStorageManager::NoteCountOptions options, QUuid requestId);
260
261     void getNoteCountPerTagComplete(
262         int noteCount, Tag tag, LocalStorageManager::NoteCountOptions options,
263         QUuid requestId);
264
265     void getNoteCountPerTagFailed(
266         ErrorString errorDescription, Tag tag,
267         LocalStorageManager::NoteCountOptions options, QUuid requestId);
268
269     void getNoteCountsPerAllTagsComplete(
270         QHash<QString, int> noteCountsPerTagLocalUids,
271         LocalStorageManager::NoteCountOptions options, QUuid requestId);
272
273     void getNoteCountsPerAllTagsFailed(
274         ErrorString errorDescription,
275         LocalStorageManager::NoteCountOptions options, QUuid requestId);
276
277     void getNoteCountPerNotebooksAndTagsComplete(
278         int noteCount, QStringList notebookLocalUids, QStringList tagLocalUids,
279         LocalStorageManager::NoteCountOptions options, QUuid requestId);
280
281     void getNoteCountPerNotebooksAndTagsFailed(
282         ErrorString errorDescription, QStringList notebookLocalUids,
283         QStringList tagLocalUids, LocalStorageManager::NoteCountOptions options,
284         QUuid requestId);
285
286     void addNoteComplete(Note note, QUuid requestId);
287
288     void addNoteFailed(
289         Note note, ErrorString errorDescription, QUuid requestId);
290
291     void updateNoteComplete(
292         Note note, LocalStorageManager::UpdateNoteOptions options,
293         QUuid requestId);
294
295     void updateNoteFailed(
296         Note note, LocalStorageManager::UpdateNoteOptions options,
297         ErrorString errorDescription, QUuid requestId);
298
299     void findNoteComplete(
300         Note foundNote, LocalStorageManager::GetNoteOptions options,
301         QUuid requestId);
302
303     void findNoteFailed(
304         Note note, LocalStorageManager::GetNoteOptions options,
305         ErrorString errorDescription, QUuid requestId);
306
307     void listNotesPerNotebookComplete(
308         Notebook notebook, LocalStorageManager::GetNoteOptions options,
309         LocalStorageManager::ListObjectsOptions flag, size_t limit,
310         size_t offset, LocalStorageManager::ListNotesOrder order,
311         LocalStorageManager::OrderDirection orderDirection,
312         QList<Note> foundNotes, QUuid requestId);
313
314     void listNotesPerNotebookFailed(
315         Notebook notebook, LocalStorageManager::GetNoteOptions options,
316         LocalStorageManager::ListObjectsOptions flag, size_t limit,
317         size_t offset, LocalStorageManager::ListNotesOrder order,
318         LocalStorageManager::OrderDirection orderDirection,
319         ErrorString errorDescription, QUuid requestId);

```

```

320
321 void listNotesPerTagComplete(
322     Tag tag, LocalStorageManager::GetNoteOptions options,
323     LocalStorageManager::ListObjectsOptions flag, size_t limit,
324     size_t offset, LocalStorageManager::ListNotesOrder order,
325     LocalStorageManager::OrderDirection orderDirection,
326     QList<Note> foundNotes, QUuid requestId);
327
328 void listNotesPerTagFailed(
329     Tag tag, LocalStorageManager::GetNoteOptions options,
330     LocalStorageManager::ListObjectsOptions flag, size_t limit,
331     size_t offset, LocalStorageManager::ListNotesOrder order,
332     LocalStorageManager::OrderDirection orderDirection,
333     ErrorString errorDescription, QUuid requestId);
334
335 void listNotesPerNotebooksAndTagsComplete(
336     QStringList notebookLocalUids, QStringList tagLocalUids,
337     LocalStorageManager::GetNoteOptions options,
338     LocalStorageManager::ListObjectsOptions flag, size_t limit,
339     size_t offset, LocalStorageManager::ListNotesOrder order,
340     LocalStorageManager::OrderDirection orderDirection,
341     QList<Note> foundNotes, QUuid requestId);
342
343 void listNotesPerNotebooksAndTagsFailed(
344     QStringList notebookLocalUids, QStringList tagLocalUids,
345     LocalStorageManager::GetNoteOptions options,
346     LocalStorageManager::ListObjectsOptions flag, size_t limit,
347     size_t offset, LocalStorageManager::ListNotesOrder order,
348     LocalStorageManager::OrderDirection orderDirection,
349     ErrorString errorDescription, QUuid requestId);
350
351 void listNotesByLocalUidsComplete(
352     QStringList noteLocalUids, LocalStorageManager::GetNoteOptions options,
353     LocalStorageManager::ListObjectsOptions flag, size_t limit,
354     size_t offset, LocalStorageManager::ListNotesOrder order,
355     LocalStorageManager::OrderDirection orderDirection,
356     QList<Note> foundNotes, QUuid requestId);
357
358 void listNotesByLocalUidsFailed(
359     QStringList noteLocalUids, LocalStorageManager::GetNoteOptions options,
360     LocalStorageManager::ListObjectsOptions flag, size_t limit,
361     size_t offset, LocalStorageManager::ListNotesOrder order,
362     LocalStorageManager::OrderDirection orderDirection,
363     ErrorString errorDescription, QUuid requestId);
364
365 void listNotesComplete(
366     LocalStorageManager::ListObjectsOptions flag,
367     LocalStorageManager::GetNoteOptions options, size_t limit,
368     size_t offset, LocalStorageManager::ListNotesOrder order,
369     LocalStorageManager::OrderDirection orderDirection,
370     QString linkedNotebookGuid, QList<Note> foundNotes, QUuid requestId);
371
372 void listNotesFailed(
373     LocalStorageManager::ListObjectsOptions flag,
374     LocalStorageManager::GetNoteOptions options, size_t limit,
375     size_t offset, LocalStorageManager::ListNotesOrder order,
376     LocalStorageManager::OrderDirection orderDirection,
377     QString linkedNotebookGuid, ErrorString errorDescription,
378     QUuid requestId);
379
380 void findNoteLocalUidsWithSearchQueryComplete(
381     QStringList noteLocalUids, NoteSearchQuery noteSearchQuery,
382     QUuid requestId);
383
384 void findNoteLocalUidsWithSearchQueryFailed(
385     NoteSearchQuery noteSearchQuery, ErrorString errorDescription,
386     QUuid requestId);
387
388 void expungeNoteComplete(Note note, QUuid requestId);
389
390 void expungeNoteFailed(
391     Note note, ErrorString errorDescription, QUuid requestId);
392
393 // Specialized signal emitted alongside updateNoteComplete (after it)
394 // if the update of a note causes the change of its notebook
395 void noteMovedToAnotherNotebook(
396     QString noteLocalUid, QString previousNotebookLocalUid,
397     QString newNotebookLocalUid);
398
399 // Specialized signal emitted alongside updateNoteComplete (after it)
400 // if the update of a note causes the change of its set of tags
401 void noteTagListChanged(
402     QString noteLocalUid, QStringList previousNoteTagLocalUids,
403     QStringList newNoteTagLocalUids);
404
405 // Tag-related signals:
406 void getTagCountComplete(int tagCount, QUuid requestId);

```

```

407 void getTagCountFailed(ErrorString errorDescription, QUuid requestId);
408 void addTagComplete(Tag tag, QUuid requestId);
409 void addTagFailed(Tag tag, ErrorString errorDescription, QUuid requestId);
410 void updateTagComplete(Tag tag, QUuid requestId);
411
412 void updateTagFailed(
413     Tag tag, ErrorString errorDescription, QUuid requestId);
414
415 void linkTagWithNoteComplete(Tag tag, Note note, QUuid requestId);
416
417 void linkTagWithNoteFailed(
418     Tag tag, Note note, ErrorString errorDescription, QUuid requestId);
419
420 void findTagComplete(Tag tag, QUuid requestId);
421 void findTagFailed(Tag tag, ErrorString errorDescription, QUuid requestId);
422
423 void listAllTagsPerNoteComplete(
424     QList<Tag> foundTags, Note note,
425     LocalStorageManager::ListObjectsOptions flag, size_t limit,
426     size_t offset, LocalStorageManager::ListTagsOrder order,
427     LocalStorageManager::OrderDirection orderDirection, QUuid requestId);
428
429 void listAllTagsPerNoteFailed(
430     Note note, LocalStorageManager::ListObjectsOptions flag, size_t limit,
431     size_t offset, LocalStorageManager::ListTagsOrder order,
432     LocalStorageManager::OrderDirection orderDirection,
433     ErrorString errorDescription, QUuid requestId);
434
435 void listAllTagsComplete(
436     size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order,
437     LocalStorageManager::OrderDirection orderDirection,
438     QString linkedNotebookGuid, QList<Tag> foundTags, QUuid requestId);
439
440 void listAllTagsFailed(
441     size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order,
442     LocalStorageManager::OrderDirection orderDirection,
443     QString linkedNotebookGuid, ErrorString errorDescription,
444     QUuid requestId);
445
446 void listTagsComplete(
447     LocalStorageManager::ListObjectsOptions flag, size_t limit,
448     size_t offset, LocalStorageManager::ListTagsOrder order,
449     LocalStorageManager::OrderDirection orderDirection,
450     QString linkedNotebookGuid, QList<Tag> foundTags,
451     QUuid requestId = QUuid());
452
453 void listTagsFailed(
454     LocalStorageManager::ListObjectsOptions flag, size_t limit,
455     size_t offset, LocalStorageManager::ListTagsOrder order,
456     LocalStorageManager::OrderDirection orderDirection,
457     QString linkedNotebookGuid, ErrorString errorDescription,
458     QUuid requestId);
459
460 void listTagsWithNoteLocalUidsComplete(
461     LocalStorageManager::ListObjectsOptions flag, size_t limit,
462     size_t offset, LocalStorageManager::ListTagsOrder order,
463     LocalStorageManager::OrderDirection orderDirection,
464     QString linkedNotebookGuid,
465     QList<std::pair<Tag, QStringList>> foundTags, QUuid requestId);
466
467 void listTagsWithNoteLocalUidsFailed(
468     LocalStorageManager::ListObjectsOptions flag, size_t limit,
469     size_t offset, LocalStorageManager::ListTagsOrder order,
470     LocalStorageManager::OrderDirection orderDirection,
471     QString linkedNotebookGuid, ErrorString errorDescription,
472     QUuid requestId);
473
474 void expungeTagComplete(
475     Tag tag, QStringList expungedChildTagLocalUids, QUuid requestId);
476
477 void expungeTagFailed(
478     Tag tag, ErrorString errorDescription, QUuid requestId);
479
480 void expungeNotelessTagsFromLinkedNotebooksComplete(QUuid requestId);
481
482 void expungeNotelessTagsFromLinkedNotebooksFailed(
483     ErrorString errorDescription, QUuid requestId);
484
485 // Resource-related signals:
486 void getResourceCountComplete(int resourceCount, QUuid requestId);
487 void getResourceCountFailed(ErrorString errorDescription, QUuid requestId);
488 void addResourceComplete(Resource resource, QUuid requestId);
489
490 void addResourceFailed(
491     Resource resource, ErrorString errorDescription, QUuid requestId);
492
493 void updateResourceComplete(Resource resource, QUuid requestId);

```

```

494
495 void updateResourceFailed(
496     Resource resource, QString errorDescription, QUuid requestId);
497
498 void findResourceComplete(
499     Resource resource, LocalStorageManager::GetResourceOptions options,
500     QUuid requestId);
501
502 void findResourceFailed(
503     Resource resource, LocalStorageManager::GetResourceOptions options,
504     QString errorDescription, QUuid requestId);
505
506 void expungeResourceComplete(Resource resource, QUuid requestId);
507
508 void expungeResourceFailed(
509     Resource resource, QString errorDescription, QUuid requestId);
510
511 // Saved search-related signals:
512 void getSavedSearchCountComplete(int savedSearchCount, QUuid requestId);
513
514 void getSavedSearchCountFailed(
515     QString errorDescription, QUuid requestId);
516
517 void addSavedSearchComplete(SavedSearch search, QUuid requestId);
518
519 void addSavedSearchFailed(
520     SavedSearch search, QString errorDescription, QUuid requestId);
521
522 void updateSavedSearchComplete(SavedSearch search, QUuid requestId);
523
524 void updateSavedSearchFailed(
525     SavedSearch search, QString errorDescription, QUuid requestId);
526
527 void findSavedSearchComplete(SavedSearch search, QUuid requestId);
528
529 void findSavedSearchFailed(
530     SavedSearch search, QString errorDescription, QUuid requestId);
531
532 void listAllSavedSearchesComplete(
533     size_t limit, size_t offset,
534     LocalStorageManager::ListSavedSearchesOrder order,
535     LocalStorageManager::OrderDirection orderDirection,
536     QList<SavedSearch> foundSearches, QUuid requestId);
537
538 void listAllSavedSearchesFailed(
539     size_t limit, size_t offset,
540     LocalStorageManager::ListSavedSearchesOrder order,
541     LocalStorageManager::OrderDirection orderDirection,
542     QString errorDescription, QUuid requestId);
543
544 void listSavedSearchesComplete(
545     LocalStorageManager::ListObjectsOptions flag, size_t limit,
546     size_t offset, LocalStorageManager::ListSavedSearchesOrder order,
547     LocalStorageManager::OrderDirection orderDirection,
548     QList<SavedSearch> foundSearches, QUuid requestId);
549
550 void listSavedSearchesFailed(
551     LocalStorageManager::ListObjectsOptions flag, size_t limit,
552     size_t offset, LocalStorageManager::ListSavedSearchesOrder order,
553     LocalStorageManager::OrderDirection orderDirection,
554     QString errorDescription, QUuid requestId);
555
556 void expungeSavedSearchComplete(SavedSearch search, QUuid requestId);
557
558 void expungeSavedSearchFailed(
559     SavedSearch search, QString errorDescription, QUuid requestId);
560
561 void accountHighUsnComplete(
562     quint32 usn, QString linkedNotebookGuid, QUuid requestId);
563
564 void accountHighUsnFailed(
565     QString linkedNotebookGuid, QString errorDescription,
566     QUuid requestId);
567
568 public Q_SLOTS:
569     void init();
570
571 // User-related slots:
572 void onGetUserCountRequest(QUuid requestId);
573
574 void onSwitchUserRequest(
575     Account account, LocalStorageManager::StartupOptions startupOptions,
576     QUuid requestId);
577
578 void onAddUserRequest(User user, QUuid requestId);
579 void onUpdateUserRequest(User user, QUuid requestId);
580 void onFindUserRequest(User user, QUuid requestId);

```

```

581 void onDeleteUserRequest(User user, QUuid requestId);
582 void onExpungeUserRequest(User user, QUuid requestId);
583
584 // Notebook-related slots:
585 void onGetNotebookCountRequest(QUuid requestId);
586 void onAddNotebookRequest(Notebook notebook, QUuid requestId);
587 void onUpdateNotebookRequest(Notebook notebook, QUuid requestId);
588 void onFindNotebookRequest(Notebook notebook, QUuid requestId);
589 void onFindDefaultNotebookRequest(Notebook notebook, QUuid requestId);
590 void onFindLastUsedNotebookRequest(Notebook notebook, QUuid requestId);
591
592 void onFindDefaultOrLastUsedNotebookRequest(
593     Notebook notebook, QUuid requestId);
594
595 void onListAllNotebooksRequest(
596     size_t limit, size_t offset,
597     LocalStorageManager::ListNotebooksOrder order,
598     LocalStorageManager::OrderDirection orderDirection,
599     QString linkedNotebookGuid, QUuid requestId);
600
601 void onListAllSharedNotebooksRequest(QUuid requestId);
602
603 void onListNotebooksRequest(
604     LocalStorageManager::ListObjectsOptions flag, size_t limit,
605     size_t offset, LocalStorageManager::ListNotebooksOrder order,
606     LocalStorageManager::OrderDirection orderDirection,
607     QString linkedNotebookGuid, QUuid requestId);
608
609 void onListSharedNotebooksPerNotebookGuidRequest(
610     QString notebookGuid, QUuid requestId);
611
612 void onExpungeNotebookRequest(Notebook notebook, QUuid requestId);
613
614 // Linked notebook-related slots:
615 void onGetLinkedNotebookCountRequest(QUuid requestId);
616
617 void onAddLinkedNotebookRequest(
618     LinkedNotebook linkedNotebook, QUuid requestId);
619
620 void onUpdateLinkedNotebookRequest(
621     LinkedNotebook linkedNotebook, QUuid requestId);
622
623 void onFindLinkedNotebookRequest(
624     LinkedNotebook linkedNotebook, QUuid requestId);
625
626 void onListAllLinkedNotebooksRequest(
627     size_t limit, size_t offset,
628     LocalStorageManager::ListLinkedNotebooksOrder order,
629     LocalStorageManager::OrderDirection orderDirection, QUuid requestId);
630
631 void onListLinkedNotebooksRequest(
632     LocalStorageManager::ListObjectsOptions flag, size_t limit,
633     size_t offset, LocalStorageManager::ListLinkedNotebooksOrder order,
634     LocalStorageManager::OrderDirection orderDirection, QUuid requestId);
635
636 void onExpungeLinkedNotebookRequest(
637     LinkedNotebook linkedNotebook, QUuid requestId);
638
639 // Note-related slots:
640 void onGetNoteCountRequest(
641     LocalStorageManager::NoteCountOptions options, QUuid requestId);
642
643 void onGetNoteCountPerNotebookRequest(
644     Notebook notebook, LocalStorageManager::NoteCountOptions options,
645     QUuid requestId);
646
647 void onGetNoteCountPerTagRequest(
648     Tag tag, LocalStorageManager::NoteCountOptions options,
649     QUuid requestId);
650
651 void onGetNoteCountsPerAllTagsRequest(
652     LocalStorageManager::NoteCountOptions options, QUuid requestId);
653
654 void onGetNoteCountPerNotebooksAndTagsRequest(
655     QStringList notebookLocalUids, QStringList tagLocalUids,
656     LocalStorageManager::NoteCountOptions options, QUuid requestId);
657
658 void onAddNoteRequest(Note note, QUuid requestId);
659
660 void onUpdateNoteRequest(
661     Note note, LocalStorageManager::UpdateNoteOptions options,
662     QUuid requestId);
663
664 void onFindNoteRequest(
665     Note note, LocalStorageManager::GetNoteOptions options,
666     QUuid requestId);
667

```

```

668 void onListNotesPerNotebookRequest(
669     Notebook notebook, LocalStorageManager::GetNoteOptions options,
670     LocalStorageManager::ListObjectsOptions flag, size_t limit,
671     size_t offset, LocalStorageManager::ListNotesOrder order,
672     LocalStorageManager::OrderDirection orderDirection, QUuid requestId);
673
674 void onListNotesPerTagRequest(
675     Tag tag, LocalStorageManager::GetNoteOptions options,
676     LocalStorageManager::ListObjectsOptions flag, size_t limit,
677     size_t offset, LocalStorageManager::ListNotesOrder order,
678     LocalStorageManager::OrderDirection orderDirection, QUuid requestId);
679
680 void onListNotesPerNotebooksAndTagsRequest(
681     QStringList notebookLocalUids, QStringList tagLocalUids,
682     LocalStorageManager::GetNoteOptions options,
683     LocalStorageManager::ListObjectsOptions flag, size_t limit,
684     size_t offset, LocalStorageManager::ListNotesOrder order,
685     LocalStorageManager::OrderDirection orderDirection, QUuid requestId);
686
687 void onListNotesByLocalUidsRequest(
688     QStringList noteLocalUids, LocalStorageManager::GetNoteOptions options,
689     LocalStorageManager::ListObjectsOptions flag, size_t limit,
690     size_t offset, LocalStorageManager::ListNotesOrder order,
691     LocalStorageManager::OrderDirection orderDirection, QUuid requestId);
692
693 void onListNotesRequest(
694     LocalStorageManager::ListObjectsOptions flag,
695     LocalStorageManager::GetNoteOptions options, size_t limit,
696     size_t offset, LocalStorageManager::ListNotesOrder order,
697     LocalStorageManager::OrderDirection orderDirection,
698     QString linkedNotebookGuid, QUuid requestId);
699
700 void onFindNoteLocalUidsWithSearchQuery(
701     NoteSearchQuery noteSearchQuery, QUuid requestId);
702
703 void onExpungeNoteRequest(Note note, QUuid requestId);
704
705 // Tag-related slots:
706 void onGetTagCountRequest(QUuid requestId);
707 void onAddTagRequest(Tag tag, QUuid requestId);
708 void onUpdateTagRequest(Tag tag, QUuid requestId);
709 void onFindTagRequest(Tag tag, QUuid requestId);
710
711 void onListAllTagsPerNoteRequest(
712     Note note, LocalStorageManager::ListObjectsOptions flag, size_t limit,
713     size_t offset, LocalStorageManager::ListTagsOrder order,
714     LocalStorageManager::OrderDirection orderDirection, QUuid requestId);
715
716 void onListAllTagsRequest(
717     size_t limit, size_t offset, LocalStorageManager::ListTagsOrder order,
718     LocalStorageManager::OrderDirection orderDirection,
719     QString linkedNotebookGuid, QUuid requestId);
720
721 void onListTagsRequest(
722     LocalStorageManager::ListObjectsOptions flag, size_t limit,
723     size_t offset, LocalStorageManager::ListTagsOrder order,
724     LocalStorageManager::OrderDirection orderDirection,
725     QString linkedNotebookGuid, QUuid requestId);
726
727 void onListTagsWithNoteLocalUidsRequest(
728     LocalStorageManager::ListObjectsOptions flag, size_t limit,
729     size_t offset, LocalStorageManager::ListTagsOrder order,
730     LocalStorageManager::OrderDirection orderDirection,
731     QString linkedNotebookGuid, QUuid requestId);
732
733 void onExpungeTagRequest(Tag tag, QUuid requestId);
734 void onExpungeNotelessTagsFromLinkedNotebooksRequest(QUuid requestId);
735
736 // Resource-related slots:
737 void onGetResourceCountRequest(QUuid requestId);
738 void onAddResourceRequest(Resource resource, QUuid requestId);
739 void onUpdateResourceRequest(Resource resource, QUuid requestId);
740
741 void onFindResourceRequest(
742     Resource resource, LocalStorageManager::GetResourceOptions options,
743     QUuid requestId);
744
745 void onExpungeResourceRequest(Resource resource, QUuid requestId);
746
747 // Saved search-related slots:
748 void onGetSavedSearchCountRequest(QUuid requestId);
749 void onAddSavedSearchRequest(SavedSearch search, QUuid requestId);
750 void onUpdateSavedSearchRequest(SavedSearch search, QUuid requestId);
751 void onFindSavedSearchRequest(SavedSearch search, QUuid requestId);
752
753 void onListAllSavedSearchesRequest(
754     size_t limit, size_t offset,

```



```

755         LocalStorageManager::ListSavedSearchesOrder order,
756         LocalStorageManager::OrderDirection orderDirection, QUuid requestId);
757
758     void onListSavedSearchesRequest (
759         LocalStorageManager::ListObjectsOptions flag, size_t limit,
760         size_t offset, LocalStorageManager::ListSavedSearchesOrder order,
761         LocalStorageManager::OrderDirection orderDirection, QUuid requestId);
762
763     void onExpungeSavedSearchRequest (SavedSearch search, QUuid requestId);
764
765     void onAccountHighUsnRequest (QString linkedNotebookGuid, QUuid requestId);
766
767 private:
768     LocalStorageManagerAsync() = delete;
769     Q_DISABLE_COPY (LocalStorageManagerAsync)
770
771     LocalStorageManagerAsyncPrivate * const d_ptr;
772     Q_DECLARE_PRIVATE (LocalStorageManagerAsync)
773 };
774
775 } // namespace quantier
776
777 #endif // LIB_QUENTIER_LOCAL_STORAGE_LOCAL_STORAGE_MANAGER_ASYNC_H

```

## 6.23 NoteSearchQuery.h

```

1  /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_LOCAL_STORAGE_NOTE_SEARCH_QUERY_H
20 #define LIB_QUENTIER_LOCAL_STORAGE_NOTE_SEARCH_QUERY_H
21
22 #include <quantier/types/ErrorString.h>
23
24 #include <QSharedDataPointer>
25
26 namespace quantier {
27
28 QT_FORWARD_DECLARE_CLASS (NoteSearchQueryData)
29
30 class QUENTIER_EXPORT NoteSearchQuery : public Printable
31 {
32 public:
33     NoteSearchQuery();
34     NoteSearchQuery(const NoteSearchQuery & other);
35     NoteSearchQuery(NoteSearchQuery && other);
36     NoteSearchQuery & operator=(const NoteSearchQuery & other);
37     NoteSearchQuery & operator=(NoteSearchQuery && other);
38     virtual ~NoteSearchQuery();
39
40     bool isEmpty() const;
41
42     void clear();
43
44     const QString queryString() const;
45
46     bool setQueryString(const QString & queryString, ErrorString & error);
47
48     const QString notebookModifier() const;
49
50     bool hasAnyModifier() const;
51
52     const QStringList & tagNames() const;
53     const QStringList & negatedTagNames() const;
54     bool hasAnyTag() const;
55     bool hasNegatedAnyTag() const;
56
57     const QStringList & titleNames() const;

```

```
66     const QStringList & negatedTitleNames() const;
67     bool hasAnyTitleName() const;
68     bool hasNegatedAnyTitleName() const;
69
70     const QVector<qint64> & creationTimestamps() const;
71     const QVector<qint64> & negatedCreationTimestamps() const;
72     bool hasAnyCreationTimestamp() const;
73     bool hasNegatedAnyCreationTimestamp() const;
74
75     const QVector<qint64> & modificationTimestamps() const;
76     const QVector<qint64> & negatedModificationTimestamps() const;
77     bool hasAnyModificationTimestamp() const;
78     bool hasNegatedAnyModificationTimestamp() const;
79
80     const QStringList & resourceMimeTypes() const;
81     const QStringList & negatedResourceMimeTypes() const;
82     bool hasAnyResourceMimeType() const;
83     bool hasNegatedAnyResourceMimeType() const;
84
85     const QVector<qint64> & subjectDateTimestamps() const;
86     const QVector<qint64> & negatedSubjectDateTimestamps() const;
87     bool hasAnySubjectDateTimestamp() const;
88     bool hasNegatedAnySubjectDateTimestamp() const;
89
90     const QVector<double> & latitudes() const;
91     const QVector<double> & negatedLatitudes() const;
92     bool hasAnyLatitude() const;
93     bool hasNegatedAnyLatitude() const;
94
95     const QVector<double> & longitudes() const;
96     const QVector<double> & negatedLongitudes() const;
97     bool hasAnyLongitude() const;
98     bool hasNegatedAnyLongitude() const;
99
100    const QVector<double> & altitudes() const;
101    const QVector<double> & negatedAltitudes() const;
102    bool hasAnyAltitude() const;
103    bool hasNegatedAnyAltitude() const;
104
105    const QStringList & authors() const;
106    const QStringList & negatedAuthors() const;
107    bool hasAnyAuthor() const;
108    bool hasNegatedAnyAuthor() const;
109
110    const QStringList & sources() const;
111    const QStringList & negatedSources() const;
112    bool hasAnySource() const;
113    bool hasNegatedAnySource() const;
114
115    const QStringList & sourceApplications() const;
116    const QStringList & negatedSourceApplications() const;
117    bool hasAnySourceApplication() const;
118    bool hasNegatedAnySourceApplication() const;
119
120    const QStringList & contentClasses() const;
121    const QStringList & negatedContentClasses() const;
122    bool hasAnyContentClass() const;
123    bool hasNegatedAnyContentClass() const;
124
125    const QStringList & placeNames() const;
126    const QStringList & negatedPlaceNames() const;
127    bool hasAnyPlaceName() const;
128    bool hasNegatedAnyPlaceName() const;
129
130    const QStringList & applicationData() const;
131    const QStringList & negatedApplicationData() const;
132    bool hasAnyApplicationData() const;
133    bool hasNegatedAnyApplicationData() const;
134
135    const QVector<qint64> & reminderOrders() const;
136    const QVector<qint64> & negatedReminderOrders() const;
137    bool hasAnyReminderOrder() const;
138    bool hasNegatedAnyReminderOrder() const;
139
140    const QVector<qint64> & reminderTimes() const;
141    const QVector<qint64> & negatedReminderTimes() const;
142    bool hasAnyReminderTime() const;
143    bool hasNegatedAnyReminderTime() const;
144
145    const QVector<qint64> & reminderDoneTimes() const;
146    const QVector<qint64> & negatedReminderDoneTimes() const;
147    bool hasAnyReminderDoneTime() const;
148    bool hasNegatedAnyReminderDoneTime() const;
149
150    bool hasUnfinishedToDo() const;
151    bool hasNegatedUnfinishedToDo() const;
152
```

```

153     bool hasFinishedToDo() const;
154     bool hasNegatedFinishedToDo() const;
155
156     bool hasAnyToDo() const;
157     bool hasNegatedAnyToDo() const;
158
159     bool hasEncryption() const;
160     bool hasNegatedEncryption() const;
161
162     const QStringList & contentSearchTerms() const;
163     const QStringList & negatedContentSearchTerms() const;
164     bool hasAnyContentSearchTerms() const;
165
166     bool isMatcheable() const;
167
168     virtual QTextStream & print(QTextStream & strm) const override;
169
170 private:
171     QSharedDataPointer<NoteSearchQueryData> d;
172 };
173
174 } // namespace quentier
175
176 #endif // LIB_QUENTIER_LOCAL_STORAGE_NOTE_SEARCH_QUERY_H

```

## 6.24 QuentierLogger.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_LOGGING_QUENTIER_LOGGER_H
20 #define LIB_QUENTIER_LOGGING_QUENTIER_LOGGER_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QDebug>
25 #include <QRegularExpression>
26 #include <QString>
27 #include <QTextStream>
28
29 namespace quentier {
30
31     enum class LogLevel
32     {
33         Trace,
34         Debug,
35         Info,
36         Warning,
37         Error
38     };
39
40     QUENTIER_EXPORT QDebug & operator<<(QDebug & dbg, const LogLevel logLevel);
41
42     QUENTIER_EXPORT QTextStream & operator<<(
43         QTextStream & strm, const LogLevel logLevel);
44
45     void QUENTIER_EXPORT QuentierInitializeLogging();
46
47     void QUENTIER_EXPORT QuentierAddLogEntry(
48         const QString & sourceFileName, const int sourceFileLineNumber,
49         const QString & component, const QString & message,
50         const LogLevel logLevel);
51
52     LogLevel QUENTIER_EXPORT QuentierMinLogLevel();
53
54     void QUENTIER_EXPORT QuentierSetMinLogLevel(const LogLevel logLevel);
55
56     void QUENTIER_EXPORT QuentierAddStdOutLogDestination();
57

```

```

83
88 bool QUENTIER_EXPORT QuantierIsLogLevelActive(const LogLevel logLevel);
89
93 QString QUENTIER_EXPORT QuantierLogFilesDirPath();
94
98 void QUENTIER_EXPORT QuantierRestartLogging();
99
103 QRegularExpression QUENTIER_EXPORT QuantierLogComponentFilter();
104
108 void QUENTIER_EXPORT
109 QuantierSetLogComponentFilter(const QRegularExpression & filter);
110
111 } // namespace quantier
112
113 #define __QNLOG_BASE(component, message, level)
114 if (quantier::QuantierIsLogLevelActive(quantier::LogLevel::level)) {
115     QString msg;
116     QDebug dbg(&msg);
117     dbg.nospace();
118     dbg.noquote();
119     dbg < message;
120     quantier::QuantierAddLogEntry(
121         QStringLiteral(__FILE__), __LINE__, QString::fromUtf8(component),
122         msg, quantier::LogLevel::level);
123 }
124 // __QNLOG_BASE
125
126 #define QNTRACE(component, message)
127 __QNLOG_BASE(component, message, Trace)
128 // QNTRACE
129
130 #define QNDEBUG(component, message)
131 __QNLOG_BASE(component, message, Debug)
132 // QNDEBUG
133
134 #define QNINFO(component, message)
135 __QNLOG_BASE(component, message, Info)
136 // QNINFO
137
138 #define QNWARNING(component, message)
139 __QNLOG_BASE(component, message, Warning)
140 // QNWARNING
141
142 #define QNERROR(component, message)
143 __QNLOG_BASE(component, message, Error)
144 // QNERROR
145
146 #define QUENTIER_SET_MIN_LOG_LEVEL(level)
147 quantier::QuantierSetMinLogLevel(
148     quantier::LogLevel::level) // QUENTIER_SET_MIN_LOG_LEVEL
149
150 #define QUENTIER_INITIALIZE_LOGGING()
151 quantier::QuantierInitializeLogging() // QUENTIER_INITIALIZE_LOGGING
152
153 #define QUENTIER_ADD_STDOUT_LOG_DESTINATION()
154 quantier::
155     QuantierAddStdOutLogDestination() // QUENTIER_ADD_STDOUT_LOG_DESTINATION
156
157 #define QNLOG_FILE_LINENUMBER_DELIMITER ":"
158
159 #endif // LIB_QUENTIER_LOGGING_QUENTIER_LOGGER_H

```

## 6.25 INoteEditorBackend.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18

```

```

19 #ifndef LIB_QUENTIER_NOTE_EDITOR_I_NOTE_EDITOR_BACKEND_H
20 #define LIB_QUENTIER_NOTE_EDITOR_I_NOTE_EDITOR_BACKEND_H
21
22 #include <quentier/types/Note.h>
23 #include <quentier/utility/Linkage.h>
24 #include <quentier/utility/Printable.h>
25
26 #include <QPalette>
27 #include <QPrinter>
28 #include <QStringList>
29 #include <QThread>
30 #include <QWidget>
31
32 QT_FORWARD_DECLARE_CLASS(QUndoStack)
33
34 namespace quentier {
35
36 QT_FORWARD_DECLARE_CLASS(Account)
37 QT_FORWARD_DECLARE_CLASS(LocalStorageManagerAsync)
38 QT_FORWARD_DECLARE_CLASS(NoteEditor)
39 QT_FORWARD_DECLARE_CLASS(SpellChecker)
40
41 class QUENTIER_EXPORT INoteEditorBackend
42 {
43 public:
44     virtual ~INoteEditorBackend();
45
46     virtual void initialize(
47         LocalStorageManagerAsync & localStorageManager,
48         SpellChecker & spellChecker, const Account & account,
49         QThread * pBackgroundJobsThread) = 0;
50
51     virtual QObject * object() = 0; // provide QObject interface
52     virtual QWidget * widget() = 0; // provide QWidget interface
53
54     virtual void setAccount(const Account & account) = 0;
55     virtual void setUndoStack(QUndoStack * pUndoStack) = 0;
56
57     virtual void setInitialPageHtml(const QString & html) = 0;
58     virtual void setNoteNotFoundPageHtml(const QString & html) = 0;
59     virtual void setNoteDeletedPageHtml(const QString & html) = 0;
60     virtual void setNoteLoadingPageHtml(const QString & html) = 0;
61
62     virtual bool isNoteLoaded() const = 0;
63     virtual qint64 idleTime() const = 0;
64
65     virtual void convertToNote() = 0;
66     virtual void saveNoteToLocalStorage() = 0;
67     virtual void setNoteTitle(const QString & noteTitle) = 0;
68
69     virtual void setTagIds(
70         const QStringList & tagLocalUids, const QStringList & tagGuids) = 0;
71
72     virtual void undo() = 0;
73     virtual void redo() = 0;
74     virtual void cut() = 0;
75     virtual void copy() = 0;
76     virtual void paste() = 0;
77     virtual void pasteUnformatted() = 0;
78     virtual void selectAll() = 0;
79
80     virtual void formatSelectionAsSourceCode() = 0;
81
82     virtual void fontMenu() = 0;
83     virtual void textBold() = 0;
84     virtual void textItalic() = 0;
85     virtual void textUnderline() = 0;
86     virtual void textStrikethrough() = 0;
87     virtual void textHighlight() = 0;
88
89     virtual void alignLeft() = 0;
90     virtual void alignCenter() = 0;
91     virtual void alignRight() = 0;
92     virtual void alignFull() = 0;
93
94     virtual QString selectedText() const = 0;
95     virtual bool hasSelection() const = 0;
96
97     virtual void findNext(const QString & text, const bool matchCase) const = 0;
98
99     virtual void findPrevious(
100         const QString & text, const bool matchCase) const = 0;
101
102     virtual void replace(
103         const QString & textToReplace, const QString & replacementText,
104         const bool matchCase) = 0;
105

```

```

106     virtual void replaceAll(
107         const QString & textToReplace, const QString & replacementText,
108         const bool matchCase) = 0;
109
110     virtual void insertToDoCheckbox() = 0;
111
112     virtual void insertInAppNoteLink(
113         const QString & userId, const QString & shardId,
114         const QString & noteGuid, const QString & linkText) = 0;
115
116     virtual void setSpellcheck(const bool enabled) = 0;
117     virtual bool spellCheckEnabled() const = 0;
118
119     virtual void setFont(const QFont & font) = 0;
120     virtual void setFontHeight(const int height) = 0;
121     virtual void setFontColor(const QColor & color) = 0;
122     virtual void setBackgroundColor(const QColor & color) = 0;
123
124     virtual QPalette defaultPalette() const = 0;
125     virtual void setDefaultPalette(const QPalette & pal) = 0;
126
127     virtual const QFont * defaultFont() const = 0;
128     virtual void setDefaultFont(const QFont & font) = 0;
129
130     virtual void insertHorizontalLine() = 0;
131
132     virtual void increaseFontSize() = 0;
133     virtual void decreaseFontSize() = 0;
134
135     virtual void increaseIndentation() = 0;
136     virtual void decreaseIndentation() = 0;
137
138     virtual void insertBulletedList() = 0;
139     virtual void insertNumberedList() = 0;
140
141     virtual void insertTableDialog() = 0;
142
143     virtual void insertFixedWidthTable(
144         const int rows, const int columns, const int widthInPixels) = 0;
145
146     virtual void insertRelativeWidthTable(
147         const int rows, const int columns, const double relativeWidth) = 0;
148
149     virtual void insertTableRow() = 0;
150     virtual void insertTableColumn() = 0;
151     virtual void removeTableRow() = 0;
152     virtual void removeTableColumn() = 0;
153
154     virtual void addAttachmentDialog() = 0;
155     virtual void saveAttachmentDialog(const QByteArray & resourceHash) = 0;
156     virtual void saveAttachmentUnderCursor() = 0;
157     virtual void openAttachment(const QByteArray & resourceHash) = 0;
158     virtual void openAttachmentUnderCursor() = 0;
159     virtual void copyAttachment(const QByteArray & resourceHash) = 0;
160     virtual void copyAttachmentUnderCursor() = 0;
161     virtual void removeAttachment(const QByteArray & resourceHash) = 0;
162     virtual void removeAttachmentUnderCursor() = 0;
163     virtual void renameAttachment(const QByteArray & resourceHash) = 0;
164     virtual void renameAttachmentUnderCursor() = 0;
165
166     enum class Rotation
167     {
168         Clockwise = 0,
169         Counterclockwise
170     };
171
172     friend QUENTIER_EXPORT QTextStream & operator<<(
173         QTextStream & strm, const Rotation rotation);
174
175     friend QUENTIER_EXPORT QDebug & operator<<(
176         QDebug & dbg, const Rotation rotation);
177
178     virtual void rotateImageAttachment(
179         const QByteArray & resourceHash, const Rotation rotationDirection) = 0;
180
181     virtual void rotateImageAttachmentUnderCursor(
182         const Rotation rotationDirection) = 0;
183
184     virtual void encryptSelectedText() = 0;
185
186     virtual void decryptEncryptedTextUnderCursor() = 0;
187
188     virtual void decryptEncryptedText(
189         QString encryptedText, QString cipher, QString keyLength, QString hint,
190         QString encryptIndex) = 0;
191
192     virtual void hideDecryptedTextUnderCursor() = 0;

```

```

193
194     virtual void hideDecryptedText(
195         QString encryptedText, QString decryptedText, QString cipher,
196         QString keyLength, QString hint, QString encryptedIndex) = 0;
197
198     virtual void editHyperlinkDialog() = 0;
199     virtual void copyHyperlink() = 0;
200     virtual void removeHyperlink() = 0;
201
202     virtual void onNoteLoadCancelled() = 0;
203
204     virtual bool print(QPrinter & printer, ErrorString & errorDescription) = 0;
205
206     virtual bool exportToPdf(
207         const QString & absoluteFilePath, ErrorString & errorDescription) = 0;
208
209     virtual bool exportToEnex(
210         const QStringList & tagNames, QString & enex,
211         ErrorString & errorDescription) = 0;
212
213     virtual QString currentNoteLocalUid() const = 0;
214     virtual void setCurrentNoteLocalUid(const QString & noteLocalUid) = 0;
215
216     virtual void clear() = 0;
217
218     virtual bool isModified() const = 0;
219     virtual bool isEditorPageModified() const = 0;
220
221     virtual void setFocusToEditor() = 0;
222
223 protected:
224     INoteEditorBackend(NoteEditor * parent);
225     NoteEditor * m_pNoteEditor;
226 };
227
228 } // namespace quantier
229
230 #endif // LIB_QUENTIER_NOTE_EDITOR_I_NOTE_EDITOR_BACKEND_H

```

## 6.26 NoteEditor.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_NOTE_EDITOR_NOTE_EDITOR_H
20 #define LIB_QUENTIER_NOTE_EDITOR_NOTE_EDITOR_H
21
22 #include <quantier/types/ErrorString.h>
23 #include <quantier/types/Note.h>
24 #include <quantier/types/Notebook.h>
25 #include <quantier/utility/Linkage.h>
26
27 #include <QPrinter>
28 #include <QStringList>
29 #include <QThread>
30 #include <QWidget>
31
32 QT_FORWARD_DECLARE_CLASS(QUndoStack)
33
34 namespace quantier {
35
36     QT_FORWARD_DECLARE_CLASS(Account)
37     QT_FORWARD_DECLARE_CLASS(INoteEditorBackend)
38     QT_FORWARD_DECLARE_CLASS(LocalStorageManagerAsync)
39     QT_FORWARD_DECLARE_CLASS(SpellChecker)
40
41     class QUENTIER_EXPORT NoteEditor : public QWidget

```

```

46 {
47     Q_OBJECT
48 public:
49     explicit NoteEditor(
50         QWidget * parent = nullptr,
51 #if QT_VERSION >= QT_VERSION_CHECK(5, 15, 0)
52         Qt::WindowFlags flags = {});
53 #else
54         Qt::WindowFlags flags = 0);
55 #endif
56
57     virtual ~NoteEditor() override;
58
59     void initialize(
60         LocalStorageManagerAsync & localStorageManager,
61         SpellChecker & spellChecker, const Account & account,
62         QThread * pBackgroundJobsThread = nullptr);
63
64     INoteEditorBackend * backend();
65
66     void setBackend(INoteEditorBackend * backend);
67
68     void setAccount(const Account & account);
69
70     const QUndoStack * undoStack() const;
71
72     void setUndoStack(QUndoStack * pUndoStack);
73
74     void setInitialPageHtml(const QString & html);
75
76     void setNoteNotFoundPageHtml(const QString & html);
77
78     void setNoteDeletedPageHtml(const QString & html);
79
80     void setNoteLoadingPageHtml(const QString & html);
81
82     QString currentNoteLocalUid() const;
83
84     void setCurrentNoteLocalUid(const QString & noteLocalUid);
85
86     void clear();
87
88     bool isModified() const;
89
90     bool isEditorPageModified() const;
91
92     bool isNoteLoaded() const;
93
94     qint64 idleTime() const;
95
96     void setFocus();
97
98     QString selectedText() const;
99     bool hasSelection() const;
100
101     bool spellCheckEnabled() const;
102
103     bool print(QPrinter & printer, ErrorString & errorDescription);
104
105     bool exportToPdf(
106         const QString & absoluteFilePath, ErrorString & errorDescription);
107
108     bool exportToEnex(
109         const QStringList & tagNames, QString & enex,
110         ErrorString & errorDescription);
111
112     QPalette defaultPalette() const;
113
114     const QFont * defaultFont() const;
115
116 Q_SIGNALS:
117     void contentChanged();
118
119     void noteAndNotebookFoundInLocalStorage(Note note, Notebook notebook);
120
121     void noteNotFound(QString noteLocalUid);
122
123     void noteDeleted(QString noteLocalUid);
124
125     void noteModified();
126
127     void notifyError(ErrorString error);
128
129     void inAppNoteLinkClicked(
130         QString userId, QString shardId, QString noteGuid);
131
132     void inAppNoteLinkPasteRequested(

```



```

270     QString url, QString userId, QString shardId, QString noteGuid);
271
272     void convertedToNote(Note note);
273     void cantConvertToNote(ErrorString error);
274
275     void noteEditorHtmlUpdated(QString html);
276
277     void currentNoteChanged(Note note);
278
279     void spellCheckerNotReady();
280     void spellCheckerReady();
281
282     void noteLoaded();
283
284     void noteSavedToLocalStorage(QString noteLocalUid);
285
286     void failedToSaveNoteToLocalStorage(
287         ErrorString errorDescription, QString noteLocalUid);
288
289     // Signals to notify anyone interested of the formatting at the current
290     // cursor position
291     void textBoldState(bool state);
292     void textItalicState(bool state);
293     void textUnderlineState(bool state);
294     void textStrikethroughState(bool state);
295     void textAlignLeftState(bool state);
296     void textAlignCenterState(bool state);
297     void textAlignRightState(bool state);
298     void textAlignFullState(bool state);
299     void textInsideOrderedListState(bool state);
300     void textInsideUnorderedListState(bool state);
301     void textInsideTableState(bool state);
302
303     void textFontFamilyChanged(QString fontFamily);
304     void textFontSizeChanged(int fontSize);
305
306     void insertTableDialogRequested();
307
308 public Q_SLOTS:
309     void convertToNote();
310
311     void saveNoteToLocalStorage();
312
313     void setNoteTitle(const QString & noteTitle);
314
315     void setTagIds(
316         const QStringList & tagLocalUids, const QStringList & tagGuids);
317
318     void undo();
319     void redo();
320     void cut();
321     void copy();
322     void paste();
323     void pasteUnformatted();
324     void selectAll();
325
326     void formatSelectionAsSourceCode();
327
328     void fontMenu();
329     void textBold();
330     void textItalic();
331     void textUnderline();
332     void textStrikethrough();
333     void textHighlight();
334
335     void alignLeft();
336     void alignCenter();
337     void alignRight();
338     void alignFull();
339
340     void findNext(const QString & text, const bool matchCase) const;
341     void findPrevious(const QString & text, const bool matchCase) const;
342
343     void replace(
344         const QString & textToReplace, const QString & replacementText,
345         const bool matchCase);
346
347     void replaceAll(
348         const QString & textToReplace, const QString & replacementText,
349         const bool matchCase);
350
351     void insertToDoCheckbox();
352
353     void insertInAppNoteLink(
354         const QString & userId, const QString & shardId,
355         const QString & noteGuid, const QString & linkText);
356

```

```

400     void setSpellcheck(const bool enabled);
401
402     void setFont(const QFont & font);
403     void setFontHeight(const int height);
404     void setFontColor(const QColor & color);
405     void setBackgroundColor(const QColor & color);
406
422     void setDefaultPalette(const QPalette & pal);
423
429     void setDefaultFont(const QFont & font);
430
431     void insertHorizontalLine();
432
433     void increaseFontSize();
434     void decreaseFontSize();
435
436     void increaseIndentation();
437     void decreaseIndentation();
438
439     void insertBulletedList();
440     void insertNumberedList();
441
442     void insertTableDialog();
443
444     void insertFixedWidthTable(
445         const int rows, const int columns, const int widthInPixels);
446
447     void insertRelativeWidthTable(
448         const int rows, const int columns, const double relativeWidth);
449
450     void insertTableRow();
451     void insertTableColumn();
452     void removeTableRow();
453     void removeTableColumn();
454
455     void addAttachmentDialog();
456     void saveAttachmentDialog(const QByteArray & resourceHash);
457     void saveAttachmentUnderCursor();
458     void openAttachment(const QByteArray & resourceHash);
459     void openAttachmentUnderCursor();
460     void copyAttachment(const QByteArray & resourceHash);
461     void copyAttachmentUnderCursor();
462
463     void encryptSelectedText();
464     void decryptEncryptedTextUnderCursor();
465
466     void editHyperlinkDialog();
467     void copyHyperlink();
468     void removeHyperlink();
469
470     void onNoteLoadCancelled();
471
472 protected:
473     virtual void dragMoveEvent(QDragMoveEvent * pEvent) override;
474     virtual void dropEvent(QDropEvent * pEvent) override;
475
476 private:
477     INoteEditorBackend * m_backend;
478 };
479
480 } // namespace quentier
481
482 #endif // LIB_QUENTIER_NOTE_EDITOR_NOTE_EDITOR_H

```

## 6.27 SpellChecker.h

```

1  /*
2  * Copyright 2017-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */

```

```

18
19 #ifndef LIB_QUENTIER_NOTE_EDITOR_SPELL_CHECKER_H
20 #define LIB_QUENTIER_NOTE_EDITOR_SPELL_CHECKER_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QObject>
25 #include <QVector>
26
27 #include <utility>
28
29 namespace quentier {
30
31 QT_FORWARD_DECLARE_CLASS(Account)
32 QT_FORWARD_DECLARE_CLASS(FileIOProcessorAsync)
33 QT_FORWARD_DECLARE_CLASS(SpellCheckerPrivate)
34
35 class QUINTIER_EXPORT SpellChecker : public QObject
36 {
37     Q_OBJECT
38 public:
39     SpellChecker(
40         FileIOProcessorAsync * pFileIOProcessorAsync, const Account & account,
41         QObject * parent = nullptr, const QString & userDictionaryPath = {});
42
43     // The second bool in the pair indicates whether the dictionary
44     // is enabled or disabled
45     QVector<std::pair<QString, bool>> listAvailableDictionaries() const;
46
47     void setAccount(const Account & account);
48
49     void enableDictionary(const QString & language);
50     void disableDictionary(const QString & language);
51
52     bool checkSpell(const QString & word) const;
53
54     QStringList spellCorrectionSuggestions(
55         const QString & misSpelledWord) const;
56
57     void addToUserWordlist(const QString & word);
58     void removeFromUserWordList(const QString & word);
59     void ignoreWord(const QString & word);
60     void removeWord(const QString & word);
61
62     bool isReady() const;
63
64 Q_SIGNALS:
65     void ready();
66
67 private:
68     SpellCheckerPrivate * const d_ptr;
69     Q_DECLARE_PRIVATE(SpellChecker)
70 };
71
72 } // namespace quentier
73
74 #endif // LIB_QUENTIER_NOTE_EDITOR_SPELL_CHECKER_H

```

## 6.28 AuthenticationManager.h

```

1 /*
2  * Copyright 2017-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_SYNCHRONIZATION_AUTHENTICATION_MANAGER_H
20 #define LIB_QUENTIER_SYNCHRONIZATION_AUTHENTICATION_MANAGER_H
21
22 #include <quentier/synchronization/IAAuthenticationManager.h>
23

```

```

24 namespace quantier {
25
26 QT_FORWARD_DECLARE_CLASS(AuthenticationManagerPrivate)
27
28
29 class QUINTIER_EXPORT AuthenticationManager : public IAuthenticationManager
30 {
31     Q_OBJECT
32 public:
33     explicit AuthenticationManager(
34         const QString & consumerKey, const QString & consumerSecret,
35         const QString & host, QObject * parent = nullptr);
36
37     virtual ~AuthenticationManager();
38
39 public Q_SLOTS:
40     virtual void onAuthenticationRequest() override;
41
42 private:
43     AuthenticationManager() = delete;
44     Q_DISABLE_COPY(AuthenticationManager)
45
46 private:
47     AuthenticationManagerPrivate * const d_ptr;
48     Q_DECLARE_PRIVATE(AuthenticationManager)
49 };
50
51 } // namespace quantier
52
53 #endif // LIB_QUENTIER_SYNCHRONIZATION_AUTHENTICATION_MANAGER_H

```

## 6.29 synchronization/ForwardDeclarations.h

```

1 /*
2  * Copyright 2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_SYNCHRONIZATION_FORWARD_DECLARATIONS_H
20 #define LIB_QUENTIER_SYNCHRONIZATION_FORWARD_DECLARATIONS_H
21
22 #include <memory>
23
24 namespace quantier {
25
26 class IAuthenticationManager;
27
28 class INoteStore;
29 using INoteStorePtr = std::shared_ptr<INoteStore>;
30
31 class ISyncStateStorage;
32 using ISyncStateStoragePtr = std::shared_ptr<ISyncStateStorage>;
33
34 class IUserStore;
35 using IUserStorePtr = std::shared_ptr<IUserStore>;
36
37 } // namespace quantier
38
39 #endif // LIB_QUENTIER_SYNCHRONIZATION_FORWARD_DECLARATIONS_H

```

## 6.30 utility/ForwardDeclarations.h

```

1 /*
2  * Copyright 2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier

```

```

5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_FORWARD_DECLARATIONS_H
20 #define LIB_QUENTIER_UTILITY_FORWARD_DECLARATIONS_H
21
22 #include <memory>
23
24 namespace quentier {
25
26 class IKeychainService;
27 using IKeychainServicePtr = std::shared_ptr<IKeychainService>;
28
29 } // namespace quentier
30
31 #endif // LIB_QUENTIER_UTILITY_FORWARD_DECLARATIONS_H

```

## 6.31 IAuthenticationManager.h

```

1 /*
2 * Copyright 2017-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_SYNCHRONIZATION_I_AUTHENTICATION_MANAGER_H
20 #define LIB_QUENTIER_SYNCHRONIZATION_I_AUTHENTICATION_MANAGER_H
21
22 #include <quentier/synchronization/ForwardDeclarations.h>
23 #include <quentier/types/ErrorMessage.h>
24 #include <quentier/utility/Linkage.h>
25
26 #include <qt5qevercloud/QEverCloud.h>
27
28 #include <QHash>
29 #include <QList>
30 #include <QNetworkCookie>
31 #include <QObject>
32 #include <QVector>
33
34 namespace quentier {
35
36 class QUENTIER_EXPORT IAuthenticationManager : public QObject
37 {
38     Q_OBJECT
39 protected:
40     explicit IAuthenticationManager(QObject * parent = nullptr);
41
42 public:
43     virtual ~IAuthenticationManager();
44
45 Q_SIGNALS:
46     void sendAuthenticationResult(
47         bool success, qevercloud::UserID userId, QString authToken,
48         qevercloud::Timestamp authTokenExpirationTime, QString shardId,
49         QString noteStoreUrl, QString webApiUrlPrefix,
50         QList<QNetworkCookie> userStoreCookies, ErrorMessage errorDescription);
51
52 public Q_SLOTS:
53     virtual void onAuthenticationRequest() = 0;

```

```

54 };
55
56 } // namespace quentier
57
58 #endif // LIB_QUENTIER_SYNCHRONIZATION_I_AUTHENTICATION_MANAGER_H

```

## 6.32 INoteStore.h

```

1 /*
2  * Copyright 2018-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_SYNCHRONIZATION_I_NOTE_STORE_H
20 #define LIB_QUENTIER_SYNCHRONIZATION_I_NOTE_STORE_H
21
22 #include <quentier/synchronization/ForwardDeclarations.h>
23 #include <quentier/types/ErrorMessage.h>
24 #include <quentier/types/Note.h>
25 #include <quentier/types/Notebook.h>
26 #include <quentier/types/SavedSearch.h>
27 #include <quentier/types/Tag.h>
28 #include <quentier/utility/Linkage.h>
29
30 #include <qt5qevercloud/QEverCloud.h>
31
32 #include <QObject>
33
34 #include <memory>
35
36 namespace quentier {
37
38 class QUENTIER_EXPORT INoteStore : public QObject
39 {
40     Q_OBJECT
41 protected:
42     explicit INoteStore(QObject * parent = nullptr);
43
44 public:
45     virtual ~INoteStore() = default;
46
47     /*
48      * Factory method, create a new INoteStore subclass object
49      */
50     virtual INoteStore * create() const = 0;
51
52     virtual QString noteStoreUrl() const = 0;
53
54     virtual void setNoteStoreUrl(QString noteStoreUrl) = 0;
55
56     virtual void setAuthData(
57         QString authenticationToken, QList<QNetworkCookie> cookies) = 0;
58
59     virtual void stop() = 0;
60
61     virtual qint32 createNotebook(
62         Notebook & notebook, ErrorMessage & errorDescription,
63         qint32 & rateLimitSeconds, QString linkedNotebookAuthToken = {}) = 0;
64
65     virtual qint32 updateNotebook(
66         Notebook & notebook, ErrorMessage & errorDescription,
67         qint32 & rateLimitSeconds, QString linkedNotebookAuthToken = {}) = 0;
68
69     virtual qint32 createNote(
70         Note & note, ErrorMessage & errorDescription, qint32 & rateLimitSeconds,
71         QString linkedNotebookAuthToken = {}) = 0;
72
73     virtual qint32 updateNote(
74         Note & note, ErrorMessage & errorDescription, qint32 & rateLimitSeconds,
75         QString linkedNotebookAuthToken = {}) = 0;
76

```

```

179
201 virtual qint32 createTag(
202     Tag & tag, ErrorString & errorDescription, qint32 & rateLimitSeconds,
203     QString linkedNotebookAuthToken = {}) = 0;
204
226 virtual qint32 updateTag(
227     Tag & tag, ErrorString & errorDescription, qint32 & rateLimitSeconds,
228     QString linkedNotebookAuthToken = {}) = 0;
229
247 virtual qint32 createSavedSearch(
248     SavedSearch & savedSearch, ErrorString & errorDescription,
249     qint32 & rateLimitSeconds) = 0;
250
267 virtual qint32 updateSavedSearch(
268     SavedSearch & savedSearch, ErrorString & errorDescription,
269     qint32 & rateLimitSeconds) = 0;
270
287 virtual qint32 getSyncState(
288     qevercloud::SyncState & syncState, ErrorString & errorDescription,
289     qint32 & rateLimitSeconds) = 0;
290
313 virtual qint32 getSyncChunk(
314     const qint32 afterUSN, const qint32 maxEntries,
315     const qevercloud::SyncChunkFilter & filter,
316     qevercloud::SyncChunk & syncChunk, ErrorString & errorDescription,
317     qint32 & rateLimitSeconds) = 0;
318
343 virtual qint32 getLinkedNotebookSyncState(
344     const qevercloud::LinkedNotebook & linkedNotebook,
345     const QString & authToken, qevercloud::SyncState & syncState,
346     ErrorString & errorDescription, qint32 & rateLimitSeconds) = 0;
347
384 virtual qint32 getLinkedNotebookSyncChunk(
385     const qevercloud::LinkedNotebook & linkedNotebook,
386     const qint32 afterUSN, const qint32 maxEntries,
387     const QString & linkedNotebookAuthToken, const bool fullSyncOnly,
388     qevercloud::SyncChunk & syncChunk, ErrorString & errorDescription,
389     qint32 & rateLimitSeconds) = 0;
390
421 virtual qint32 getNote(
422     const bool withContent, const bool withResourcesData,
423     const bool withResourcesRecognition,
424     const bool withResourceAlternateData, Note & note,
425     ErrorString & errorDescription, qint32 & rateLimitSeconds) = 0;
426
464 virtual bool getNoteAsync(
465     const bool withContent, const bool withResourceData,
466     const bool withResourcesRecognition,
467     const bool withResourceAlternateData, const bool withSharedNotes,
468     const bool withNoteAppDataValues, const bool withResourceAppDataValues,
469     const bool withNoteLimits, const QString & noteGuid,
470     const QString & authToken, ErrorString & errorDescription) = 0;
471
503 virtual qint32 getResource(
504     const bool withDataBody, const bool withRecognitionDataBody,
505     const bool withAlternateDataBody, const bool withAttributes,
506     const QString & authToken, Resource & resource,
507     ErrorString & errorDescription, qint32 & rateLimitSeconds) = 0;
508
533 virtual bool getResourceAsync(
534     const bool withDataBody, const bool withRecognitionDataBody,
535     const bool withAlternateDataBody, const bool withAttributes,
536     const QString & resourceGuid, const QString & authToken,
537     ErrorString & errorDescription) = 0;
538
558 virtual qint32 authenticateToSharedNotebook(
559     const QString & shareKey, qevercloud::AuthenticationResult & authResult,
560     ErrorString & errorDescription, qint32 & rateLimitSeconds) = 0;
561
562 Q_SIGNALS:
563     void getNoteAsyncFinished(
564         qint32 errorCode, qevercloud::Note note, qint32 rateLimitSeconds,
565         ErrorString errorDescription);
566
567     void getResourceAsyncFinished(
568         qint32 errorCode, qevercloud::Resource resource,
569         qint32 rateLimitSeconds, ErrorString errorDescription);
570
571 private:
572     Q_DISABLE_COPY(INoteStore)
573 };
574
575 QUINTIER_EXPORT INoteStorePtr newNoteStore(QObject * parent = nullptr);
576
577 } // namespace quantier
578
579 #endif // LIB_QUINTIER_SYNCHRONIZATION_I_NOTE_STORE_H

```

## 6.33 ISyncStateStorage.h

```

1  /*
2  * Copyright 2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_SYNCHRONIZATION_I_SYNC_STATE_STORAGE_H
20 #define LIB_QUENTIER_SYNCHRONIZATION_I_SYNC_STATE_STORAGE_H
21
22 #include <quentier/synchronization/ForwardDeclarations.h>
23 #include <quentier/types/Account.h>
24 #include <quentier/utility/Linkage.h>
25
26 #include <qt5qevercloud/QEverCloud.h>
27
28 #include <QHash>
29 #include <QObject>
30 #include <QString>
31
32 #include <memory>
33
34 namespace quentier {
35
41 class QUENTIER_EXPORT ISyncStateStorage : public QObject
42 {
43     Q_OBJECT
44 public:
49     class QUENTIER_EXPORT ISyncState : public Printable
50     {
51     public:
52         virtual quint32 userDataUpdateCount() const = 0;
53         virtual qevercloud::Timestamp userDataLastSyncTime() const = 0;
54         virtual QHash<QString, quint32> linkedNotebookUpdateCounts() const = 0;
55
56         virtual QHash<QString, qevercloud::Timestamp>
57             linkedNotebookLastSyncTimes() const = 0;
58
59         virtual QTextStream & print(QTextStream & strm) const override;
60     };
61
62     using ISyncStatePtr = std::shared_ptr<ISyncState>;
63
64 public:
65     explicit ISyncStateStorage(QObject * parent = nullptr) : QObject(parent) {}
66
67     virtual ~ISyncStateStorage() = default;
68
69     virtual ISyncStatePtr getSyncState(const Account & account) = 0;
70
71     virtual void setSyncState(
72         const Account & account, ISyncStatePtr syncState) = 0;
73
74     Q_SIGNALS:
75     void notifySyncStateUpdated(Account account, ISyncStatePtr syncState);
76 };
77
78 QUENTIER_EXPORT ISyncStateStoragePtr
79 newSyncStateStorage(QObject * parent = nullptr);
80
81 } // namespace quentier
82
83 #endif // LIB_QUENTIER_SYNCHRONIZATION_I_SYNC_STATE_STORAGE_H

```

## 6.34 IUserStore.h

```

1  /*
2  * Copyright 2018-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier

```



```

5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_SYNCHRONIZATION_I_USER_STORE_H
20 #define LIB_QUENTIER_SYNCHRONIZATION_I_USER_STORE_H
21
22 #include <quentier/synchronization/ForwardDeclarations.h>
23 #include <quentier/types/ErrorMessage.h>
24 #include <quentier/utility/Linkage.h>
25
26 #include <QList>
27 #include <QNetworkCookie>
28
29 #include <qt5qevercloud/QEverCloud.h>
30
31 #include <memory>
32
33 namespace quentier {
34
35 QT_FORWARD_DECLARE_CLASS(User)
36
37
38 class QUENTIER_EXPORT IUserStore
39 {
40 public:
41     virtual ~IUserStore() = default;
42
43     virtual void setAuthData(
44         QString authenticationToken, QList<QNetworkCookie> cookies) = 0;
45
46     virtual bool checkVersion(
47         const QString & clientName, qint16 edamVersionMajor,
48         qint16 edamVersionMinor, ErrorMessage & errorDescription) = 0;
49
50     virtual qint32 getUser(
51         User & user, ErrorMessage & errorDescription,
52         qint32 & rateLimitSeconds) = 0;
53
54     virtual qint32 getAccountLimits(
55         const qevercloud::ServiceLevel serviceLevel,
56         qevercloud::AccountLimits & limits, ErrorMessage & errorDescription,
57         qint32 & rateLimitSeconds) = 0;
58 };
59
60 QUENTIER_EXPORT IUserStorePtr newUserStore(QString evernoteHost);
61
62 } // namespace quentier
63
64 #endif // LIB_QUENTIER_SYNCHRONIZATION_I_USER_STORE_H

```

## 6.35 SynchronizationManager.h

```

1 /*
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_SYNCHRONIZATION_SYNCHRONIZATION_MANAGER_H
20 #define LIB_QUENTIER_SYNCHRONIZATION_SYNCHRONIZATION_MANAGER_H

```

```

21
22 #include <quentier/synchronization/ForwardDeclarations.h>
23 #include <quentier/types/Account.h>
24 #include <quentier/types/ErrorMessage.h>
25 #include <quentier/types/LinkedNotebook.h>
26 #include <quentier/utility/ForwardDeclarations.h>
27 #include <quentier/utility/Linkage.h>
28
29 #include <QObject>
30
31 namespace quentier {
32
33 QT_FORWARD_DECLARE_CLASS(LocalStorageManagerAsync)
34 QT_FORWARD_DECLARE_CLASS(SynchronizationManagerPrivate)
35
36
37 class QUINTIER_EXPORT SynchronizationManager : public QObject
38 {
39     Q_OBJECT
40 public:
41     SynchronizationManager(
42         QString host, LocalStorageManagerAsync & localStorageManagerAsync,
43         IAuthenticationManager & authenticationManager,
44         QObject * parent = nullptr, INoteStorePtr pNoteStore = {},
45         IUserStorePtr pUserStore = {},
46         IKeychainServicePtr pKeychainService = {},
47         ISyncStateStoragePtr pSyncStateStorage = {});
48
49     virtual ~SynchronizationManager();
50
51     bool active() const;
52
53     bool downloadNoteThumbnailsOption() const;
54
55 public Q_SLOTS:
56     void setAccount(Account account);
57
58     void authenticate();
59
60     void authenticateCurrentAccount();
61
62     void synchronize();
63
64     void stop();
65
66     void revokeAuthentication(const qevercloud::UserID userId);
67
68     void setDownloadNoteThumbnails(bool flag);
69
70     void setDownloadInkNoteImages(bool flag);
71
72     void setInkNoteImagesStoragePath(QString path);
73
74 Q_SIGNALS:
75     void started();
76
77     void stopped();
78
79     void failed(ErrorMessage errorDescription);
80
81     void finished(
82         Account account, bool somethingDownloaded, bool somethingSent);
83
84     void authenticationRevoked(
85         bool success, ErrorMessage errorDescription, qevercloud::UserID userId);
86
87     void authenticationFinished(
88         bool success, ErrorMessage errorDescription, Account account);
89
90     void remoteToLocalSyncStopped();
91
92     void sendLocalChangesStopped();
93
94     void willRepeatRemoteToLocalSyncAfterSendingChanges();
95
96     void detectedConflictDuringLocalChangesSending();
97
98     void rateLimitExceeded(qint32 secondsToWait);
99
100    void remoteToLocalSyncDone(bool somethingDownloaded);
101
102    void syncChunksDownloadProgress(
103        qint32 highestDownloadedUsn, qint32 highestServerUsn,
104        qint32 lastPreviousUsn);
105
106    void syncChunksDownloaded();
107

```

```

393 void linkedNotebookSyncChunksDownloadProgress(
394     qint32 highestDownloadedUsn, qint32 highestServerUsn,
395     qint32 lastPreviousUsn, LinkedNotebook linkedNotebook);
396
401 void linkedNotebooksSyncChunksDownloaded();
402
411 void notesDownloadProgress(
412     quint32 notesDownloaded, quint32 totalNotesToDownload);
413
422 void linkedNotebooksNotesDownloadProgress(
423     quint32 notesDownloaded, quint32 totalNotesToDownload);
424
435 void resourcesDownloadProgress(
436     quint32 resourcesDownloaded, quint32 totalResourcesToDownload);
437
448 void linkedNotebooksResourcesDownloadProgress(
449     quint32 resourcesDownloaded, quint32 totalResourcesToDownload);
450
456 void preparedDirtyObjectsForSending();
457
463 void preparedLinkedNotebooksDirtyObjectsForSending();
464
470 void setAccountDone(Account account);
471
476 void setDownloadNoteThumbnailsDone(bool flag);
477
482 void setDownloadInkNoteImagesDone(bool flag);
483
488 void setInkNoteImagesStoragePathDone(QString path);
489
490 private:
491     SynchronizationManager() = delete;
492     Q_DISABLE_COPY(SynchronizationManager)
493
494     SynchronizationManagerPrivate * d_ptr;
495     Q_DECLARE_PRIVATE(SynchronizationManager)
496 };
497
498 } // namespace quantier
499
500 #endif // LIB_QUENTIER_SYNCHRONIZATION_SYNCHRONIZATION_MANAGER_H

```

## 6.36 Account.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_ACCOUNT_H
20 #define LIB_QUENTIER_TYPES_ACCOUNT_H
21
22 #include <quentier/utility/Printable.h>
23
24 #include <qt5qevercloud/QEverCloud.h>
25
26 #include <QSharedDataPointer>
27 #include <QString>
28
29 namespace quantier {
30
31 QT_FORWARD_DECLARE_CLASS(AccountData)
32
33
34 class QUENTIER_EXPORT Account : public Printable
35 {
36 public:
37     enum class Type
38     {
39         Local = 0,

```

```

44     Evernote
45 };
46
47 friend QUINTIER_EXPORT QTextStream & operator<<
48     (QTextStream & strm, const Type type);
49
50 friend QUINTIER_EXPORT QDebug & operator<<(QDebug & dbg, const Type type);
51
52 enum class EvernoteAccountType
53 {
54     Free = 0,
55     Plus,
56     Premium,
57     Business
58 };
59
60 friend QUINTIER_EXPORT QTextStream & operator<<
61     (QTextStream & strm, const EvernoteAccountType type);
62
63 friend QUINTIER_EXPORT QDebug & operator<<
64     (QDebug & dbg, const EvernoteAccountType type);
65
66 public:
67     explicit Account();
68
69     explicit Account(
70         QString name, const Type type, const qevercloud::UserID userId = -1,
71         const EvernoteAccountType evernoteAccountType =
72             EvernoteAccountType::Free,
73         QString evernoteHost = {}, QString shardId = {});
74
75     Account(const Account & other);
76     Account & operator=(const Account & other);
77     virtual ~Account() override;
78
79     bool operator==(const Account & other) const;
80     bool operator!=(const Account & other) const;
81
82     bool isEmpty() const;
83
84     QString name() const;
85
86     void setName(QString name);
87
88     QString displayName() const;
89
90     void setDisplayName(QString displayName);
91
92     Type type() const;
93
94     qevercloud::UserID id() const;
95
96     EvernoteAccountType evernoteAccountType() const;
97
98     QString evernoteHost() const;
99
100    QString shardId() const;
101
102    void setEvernoteAccountType(const EvernoteAccountType evernoteAccountType);
103    void setEvernoteHost(QString evernoteHost);
104    void setShardId(QString shardId);
105
106    qint32 mailLimitDaily() const;
107    qint64 noteSizeMax() const;
108    qint64 resourceSizeMax() const;
109    qint32 linkedNotebookMax() const;
110    qint32 noteCountMax() const;
111    qint32 notebookCountMax() const;
112    qint32 tagCountMax() const;
113    qint32 noteTagCountMax() const;
114    qint32 savedSearchCountMax() const;
115    qint32 noteResourceCountMax() const;
116    void setEvernoteAccountLimits(const qevercloud::AccountLimits & limits);
117
118    virtual QTextStream & print(QTextStream & strm) const override;
119
120 private:
121     QSharedDataPointer<AccountData> d;
122 };
123
124 } // namespace quotientier
125
126 #endif // LIB_QUINTIER_TYPES_ACCOUNT_H

```

## 6.37 ErrorString.h

```

1  /*
2  * Copyright 2017-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_ERROR_STRING_H
20 #define LIB_QUENTIER_TYPES_ERROR_STRING_H
21
22 #include <quentier/utility/Printable.h>
23
24 #include <QSharedDataPointer>
25
26 namespace quentier {
27
28 QT_FORWARD_DECLARE_CLASS(ErrorStringData)
29
30
31 class QUINTIER_EXPORT ErrorString : public Printable
32 {
33 public:
34     explicit ErrorString(const char * error = nullptr);
35     explicit ErrorString(const QString & error);
36     ErrorString(const ErrorString & other);
37     ErrorString & operator=(const ErrorString & other);
38     virtual ~ErrorString() override;
39
40     const QString & base() const;
41     QString & base();
42
43     const QStringList & additionalBases() const;
44     QStringList & additionalBases();
45
46     const QString & details() const;
47     QString & details();
48
49     void setBase(const QString & error);
50     void setBase(const char * error);
51
52     void appendBase(const QString & error);
53     void appendBase(const QStringList & errors);
54     void appendBase(const char * error);
55
56     void setDetails(const QString & error);
57     void setDetails(const char * error);
58
59     bool isEmpty() const;
60     void clear();
61
62     QString localizedString() const;
63     QString nonLocalizedString() const;
64
65     virtual QTextStream & print(QTextStream & strm) const override;
66
67 private:
68     QSharedDataPointer<ErrorStringData> d;
69 };
70
71 } // namespace quentier
72
73 #endif // LIB_QUENTIER_TYPES_ERROR_STRING_H

```

## 6.38 IFavoritableDataElement.h

```

1  /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *

```

```

6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_I_FAVORITABLE_DATA_ELEMENT_H
20 #define LIB_QUENTIER_TYPES_I_FAVORITABLE_DATA_ELEMENT_H
21
22 #include "INoteStoreDataElement.h"
23
24 namespace quentier {
25
26 class QUENTIER_EXPORT IFavoritableDataElement :
27     public virtual INoteStoreDataElement
28 {
29 public:
30     virtual bool isFavorited() const = 0;
31     virtual void setFavorited(const bool favorited) = 0;
32
33     virtual ~IFavoritableDataElement() {}
34 };
35
36 #define DECLARE_IS_FAVORITED \
37     virtual bool isFavorited() const override; \
38     // DECLARE_IS_FAVORITED
39
40 #define DECLARE_SET_FAVORITED \
41     virtual void setFavorited(const bool favorited) override; \
42     // DECLARE_SET_FAVORITED
43
44 #define QN_DECLARE_FAVORITED \
45     DECLARE_IS_FAVORITED \
46     DECLARE_SET_FAVORITED \
47     // QN_DECLARE_FAVORITED
48
49 #define DEFINE_IS_FAVORITED(type) \
50     bool type::isFavorited() const \
51     { \
52         return d->m_isFavorited; \
53     } \
54     // DEFINE_IS_FAVORITED
55
56 #define DEFINE_SET_FAVORITED(type) \
57     void type::setFavorited(const bool favorited) \
58     { \
59         d->m_isFavorited = favorited; \
60     } \
61     // DEFINE_SET_FAVORITED
62
63 #define QN_DEFINE_FAVORITED(type) \
64     DEFINE_IS_FAVORITED(type) \
65     DEFINE_SET_FAVORITED(type) \
66     // QN_DEFINE_FAVORITED
67
68 } // namespace quentier
69
70 #endif // LIB_QUENTIER_TYPES_I_FAVORITABLE_DATA_ELEMENT_H

```

## 6.39 ILocalStorageDataElement.h

```

1 /*
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License

```

```

16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_I_LOCAL_STORAGE_DATA_ELEMENT_H
20 #define LIB_QUENTIER_TYPES_I_LOCAL_STORAGE_DATA_ELEMENT_H
21
22 #include <quentier/utility/Linkage.h>
23 #include <quentier/utility/UidGenerator.h>
24
25 #include <QString>
26 #include <QUuid>
27
28 namespace quentier {
29
30 class QUENTIER_EXPORT ILocalStorageDataElement
31 {
32 public:
33     virtual const QString localUid() const = 0;
34     virtual void setLocalUid(const QString & guid) = 0;
35     virtual void unsetLocalUid() = 0;
36
37     virtual ~ILocalStorageDataElement() {}
38 };
39
40 #define DEFINE_LOCAL_UID_GETTER(type)
41 const QString type::localUid() const
42 {
43     return UidGenerator::UidToString(d->m_localUid);
44 }
45 // DEFINE_LOCAL_UID_GETTER
46
47 #define DEFINE_LOCAL_UID_SETTER(type)
48 void type::setLocalUid(const QString & uid)
49 {
50     d->m_localUid = uid;
51 }
52 // DEFINE_LOCAL_UID_SETTER
53
54 #define DEFINE_LOCAL_UID_UNSETTER(type)
55 void type::unsetLocalUid()
56 {
57     d->m_localUid = QUuid();
58 }
59 // DEFINE_LOCAL_UID_UNSETTER
60
61 #define QN_DECLARE_LOCAL_UID
62 virtual const QString localUid() const override;
63 virtual void setLocalUid(const QString & guid) override;
64 virtual void unsetLocalUid() override;
65 // QN_DECLARE_LOCAL_UID
66
67 #define QN_DEFINE_LOCAL_UID(type)
68 DEFINE_LOCAL_UID_GETTER(type)
69 DEFINE_LOCAL_UID_SETTER(type)
70 DEFINE_LOCAL_UID_UNSETTER(type)
71 // QN_DEFINE_LOCAL_UID
72
73 } // namespace quentier
74
75 #endif // LIB_QUENTIER_TYPES_I_LOCAL_STORAGE_DATA_ELEMENT_H

```

## 6.40 INoteStoreDataElement.h

```

1 /*
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_I_NOTE_STORE_DATA_ELEMENT_H
20 #define LIB_QUENTIER_TYPES_I_NOTE_STORE_DATA_ELEMENT_H

```

```

21
22 #include "ILocalStorageDataElement.h"
23
24 #include <quentier/types/ErrorString.h>
25 #include <quentier/utility/Printable.h>
26
27 #include <QUuid>
28 #include <QtGlobal>
29
30 namespace quentier {
31
32 class QUENTIER_EXPORT INoteStoreDataElement :
33     public ILocalStorageDataElement,
34     public Printable
35 {
36 public:
37     virtual void clear() = 0;
38
39     virtual bool hasGuid() const = 0;
40     virtual const QString & guid() const = 0;
41     virtual void setGuid(const QString & guid) = 0;
42
43     virtual bool hasUpdateSequenceNumber() const = 0;
44     virtual quint32 updateSequenceNumber() const = 0;
45     virtual void setUpdateSequenceNumber(const quint32 usn) = 0;
46
47     virtual bool checkParameters(ErrorString & errorDescription) const = 0;
48
49     virtual bool isDirty() const = 0;
50     virtual void setDirty(const bool dirty) = 0;
51
52     virtual bool isLocal() const = 0;
53     virtual void setLocal(const bool local) = 0;
54
55     virtual ~INoteStoreDataElement() {}
56 };
57
58 #define DECLARE_IS_DIRTY
59 virtual bool isDirty() const override;
60 // DECLARE_IS_DIRTY
61
62 #define DECLARE_SET_DIRTY
63 virtual void setDirty(const bool isDirty) override;
64 // DECLARE_SET_DIRTY
65
66 #define QN_DECLARE_DIRTY
67 DECLARE_IS_DIRTY
68 DECLARE_SET_DIRTY
69 // QN_DECLARE_DIRTY
70
71 #define DEFINE_IS_DIRTY(type)
72 bool type::isDirty() const
73 {
74     return d->m_isDirty;
75 }
76 // DEFINE_IS_DIRTY
77
78 #define DEFINE_SET_DIRTY(type)
79 void type::setDirty(const bool dirty)
80 {
81     d->m_isDirty = dirty;
82 }
83 // DEFINE_SET_DIRTY
84
85 #define QN_DEFINE_DIRTY(type)
86 DEFINE_IS_DIRTY(type)
87 DEFINE_SET_DIRTY(type)
88 // QN_DEFINE_DIRTY
89
90 #define DECLARE_IS_LOCAL
91 virtual bool isLocal() const override;
92 // DECLARE_IS_LOCAL
93
94 #define DECLARE_SET_LOCAL
95 virtual void setLocal(const bool isLocal) override;
96 // DECLARE_SET_LOCAL
97
98 #define QN_DECLARE_LOCAL
99 DECLARE_IS_LOCAL
100 DECLARE_SET_LOCAL
101 // QN_DECLARE_LOCAL
102
103 #define DEFINE_IS_LOCAL(type)
104 bool type::isLocal() const
105 {
106     return d->m_isLocal;
107 }

```



```

108 // DEFINE_IS_LOCAL
109
110 #define DEFINE_SET_LOCAL(type)
111 void type::setLocal(const bool local)
112 {
113     d->m_isLocal = local;
114 }
115 // DEFINE_SET_LOCAL
116
117 #define QN_DEFINE_LOCAL(type)
118 DEFINE_IS_LOCAL(type)
119 DEFINE_SET_LOCAL(type)
120 // QN_DEFINE_LOCAL
121
122 } // namespace quantier
123
124 #endif // LIB_QUENTIER_TYPES_I_NOTE_STORE_DATA_ELEMENT_H

```

## 6.41 LinkedNotebook.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_LINKED_NOTEBOOK_H
20 #define LIB_QUENTIER_TYPES_LINKED_NOTEBOOK_H
21
22 #include "INoteStoreDataElement.h"
23
24 #include <qt5qevercloud/QEverCloud.h>
25
26 #include <QSharedDataPointer>
27
28 namespace quantier {
29
30     QT_FORWARD_DECLARE_CLASS(LinkedNotebookData)
31
32     class QUENTIER_EXPORT LinkedNotebook : public INoteStoreDataElement
33     {
34     public:
35         QN_DECLARE_DIRTY
36
37     public:
38         explicit LinkedNotebook();
39         LinkedNotebook(const LinkedNotebook & other);
40         LinkedNotebook(LinkedNotebook && other);
41         LinkedNotebook & operator=(const LinkedNotebook & other);
42         LinkedNotebook & operator=(LinkedNotebook && other);
43
44         explicit LinkedNotebook(const qevercloud::LinkedNotebook & linkedNotebook);
45         explicit LinkedNotebook(qevercloud::LinkedNotebook && linkedNotebook);
46
47         virtual ~LinkedNotebook() override;
48
49         const qevercloud::LinkedNotebook & qevercloudLinkedNotebook() const;
50         qevercloud::LinkedNotebook & qevercloudLinkedNotebook();
51
52         bool operator==(const LinkedNotebook & other) const;
53         bool operator!=(const LinkedNotebook & other) const;
54
55         virtual void clear() override;
56
57         virtual bool hasGuid() const override;
58         virtual const QString & guid() const override;
59         virtual void setGuid(const QString & guid) override;
60
61         virtual bool hasUpdateSequenceNumber() const override;
62         virtual qint32 updateSequenceNumber() const override;
63         virtual void setUpdateSequenceNumber(const qint32 usn) override;

```

```

64
65     virtual bool checkParameters(ErrorString & errorDescription) const override;
66
67     bool hasShareName() const;
68     const QString & shareName() const;
69     void setShareName(const QString & shareName);
70
71     bool hasUsername() const;
72     const QString & username() const;
73     void setUsername(const QString & username);
74
75     bool hasShardId() const;
76     const QString & shardId() const;
77     void setShardId(const QString & shardId);
78
79     bool hasSharedNotebookGlobalId() const;
80     const QString & sharedNotebookGlobalId() const;
81     void setSharedNotebookGlobalId(const QString & sharedNotebookGlobalId);
82
83     bool hasUri() const;
84     const QString & uri() const;
85     void setUri(const QString & uri);
86
87     bool hasNoteStoreUrl() const;
88     const QString & noteStoreUrl() const;
89     void setNoteStoreUrl(const QString & noteStoreUrl);
90
91     bool hasWebApiUrlPrefix() const;
92     const QString & webApiUrlPrefix() const;
93     void setWebApiUrlPrefix(const QString & webApiUrlPrefix);
94
95     bool hasStack() const;
96     const QString & stack() const;
97     void setStack(const QString & stack);
98
99     bool hasBusinessId() const;
100    qint32 businessId() const;
101    void setBusinessId(const qint32 businessId);
102
103    virtual QTextStream & print(QTextStream & strm) const override;
104
105 private:
106     // hide useless methods inherited from the base class from public interface
107     virtual const QString localUid() const override
108     {
109         return QString();
110     }
111     virtual void setLocalUid(const QString &) override {}
112     virtual void unsetLocalUid() override {}
113
114     virtual bool isLocal() const override
115     {
116         return false;
117     }
118     virtual void setLocal(const bool) override {}
119
120 private:
121     QSharedDataPointer<LinkedNotebookData> d;
122 };
123
124 } // namespace quotientier
125
126 Q_DECLARE_METATYPE(quotientier::LinkedNotebook)
127
128 #endif // LIB_QUENTIER_TYPES_LINKED_NOTEBOOK_H

```

## 6.42 Note.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquotientier
5  *
6  * libquotientier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquotientier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License

```

```

16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_NOTE_H
20 #define LIB_QUENTIER_TYPES_NOTE_H
21
22 #include "IFavoritableDataElement.h"
23 #include "SharedNote.h"
24
25 #include <qt5qevercloud/QEverCloud.h>
26
27 #include <QSharedDataPointer>
28
29 namespace quentier {
30
31 QT_FORWARD_DECLARE_CLASS(Resource)
32 QT_FORWARD_DECLARE_CLASS(NoteData)
33
34 class QUENTIER_EXPORT Note : public IFavoritableDataElement
35 {
36 public:
37     QN_DECLARE_LOCAL_UID
38     QN_DECLARE_DIRTY
39     QN_DECLARE_FAVORITED
40     QN_DECLARE_LOCAL
41
42 public:
43     explicit Note();
44     Note(const Note & other);
45     Note(Note && other);
46     Note & operator=(const Note & other);
47     Note & operator=(Note && other);
48
49     explicit Note(const qevercloud::Note & other);
50     Note & operator=(const qevercloud::Note & other);
51
52     virtual ~Note() override;
53
54     bool operator==(const Note & other) const;
55     bool operator!=(const Note & other) const;
56
57     const qevercloud::Note & qevercloudNote() const;
58     qevercloud::Note & qevercloudNote();
59
60     virtual bool hasGuid() const override;
61     virtual const QString & guid() const override;
62     virtual void setGuid(const QString & guid) override;
63
64     virtual bool hasUpdateSequenceNumber() const override;
65     virtual quint32 updateSequenceNumber() const override;
66     virtual void setUpdateSequenceNumber(const quint32 usn) override;
67
68     virtual void clear() override;
69
70     static bool validateTitle(
71         const QString & title, ErrorString * pErrorDescription = nullptr);
72
73     virtual bool checkParameters(ErrorString & errorDescription) const override;
74
75     bool hasTitle() const;
76     const QString & title() const;
77     void setTitle(const QString & title);
78
79     bool hasContent() const;
80     const QString & content() const;
81     void setContent(const QString & content);
82
83     bool hasContentHash() const;
84     const QByteArray & contentHash() const;
85     void setContentHash(const QByteArray & contentHash);
86
87     bool hasContentLength() const;
88     quint32 contentLength() const;
89     void setContentLength(const quint32 length);
90
91     bool hasCreationTimestamp() const;
92     quint64 creationTimestamp() const;
93     void setCreationTimestamp(const quint64 timestamp);
94
95     bool hasModificationTimestamp() const;
96     quint64 modificationTimestamp() const;
97     void setModificationTimestamp(const quint64 timestamp);
98
99     bool hasDeletionTimestamp() const;
100     quint64 deletionTimestamp() const;
101     void setDeletionTimestamp(const quint64 timestamp);
102

```

```

103     bool hasActive() const;
104     bool active() const;
105     void setActive(const bool active);
106
107     bool hasNotebookGuid() const;
108     const QString & notebookGuid() const;
109     void setNotebookGuid(const QString & guid);
110
111     bool hasNotebookLocalUid() const;
112     const QString & notebookLocalUid() const;
113     void setNotebookLocalUid(const QString & notebookLocalUid);
114
115     bool hasTagGuids() const;
116     const QStringList tagGuids() const;
117     void setTagGuids(const QStringList & guids);
118     void addTagGuid(const QString & guid);
119     void removeTagGuid(const QString & guid);
120
121     bool hasTagLocalUids() const;
122     const QStringList & tagLocalUids() const;
123     void setTagLocalUids(const QStringList & localUids);
124     void addTagLocalUid(const QString & localUid);
125     void removeTagLocalUid(const QString & localUid);
126
127     bool hasResources() const;
128     int numResources() const;
129     QList<Resource> resources() const;
130     void setResources(const QList<Resource> & resources);
131     void addResource(const Resource & resource);
132     bool updateResource(const Resource & resource);
133     bool removeResource(const Resource & resource);
134
135     bool hasNoteAttributes() const;
136     const qevercloud::NoteAttributes & noteAttributes() const;
137     qevercloud::NoteAttributes & noteAttributes();
138     void clearNoteAttributes();
139
140     bool hasSharedNotes() const;
141     QList<SharedNote> sharedNotes() const;
142     void setSharedNotes(const QList<SharedNote> & sharedNotes);
143     void addSharedNote(const SharedNote & sharedNote);
144
145     // NOTE: the shared note is recognized by its index in note
146     // in the following two methods
147     bool updateSharedNote(const SharedNote & sharedNote);
148     bool removeSharedNote(const SharedNote & sharedNote);
149
150     bool hasNoteRestrictions() const;
151     const qevercloud::NoteRestrictions & noteRestrictions() const;
152     qevercloud::NoteRestrictions & noteRestrictions();
153     void setNoteRestrictions(qevercloud::NoteRestrictions && restrictions);
154
155     bool hasNoteLimits() const;
156     const qevercloud::NoteLimits & noteLimits() const;
157     qevercloud::NoteLimits & noteLimits();
158     void setNoteLimits(qevercloud::NoteLimits && limits);
159
160     QByteArray thumbnailData() const;
161     void setThumbnailData(const QByteArray & thumbnailData);
162
163     bool isInkNote() const;
164
165     QString plainText(ErrorString * pErrorMessage = nullptr) const;
166     QStringList listOfWords(ErrorString * pErrorMessage = nullptr) const;
167
168     std::pair<QString, QStringList> plainTextAndListOfWords(
169         ErrorString * pErrorMessage = nullptr) const;
170
171     bool containsCheckedTodo() const;
172     bool containsUncheckedTodo() const;
173     bool containsTodo() const;
174     bool containsEncryption() const;
175
176     virtual QTextStream & print(QTextStream & strm) const override;
177
178 private:
179     QSharedDataPointer<NoteData> d;
180 };
181
182 } // namespace quentier
183
184 Q_DECLARE_METATYPE(quentier::Note)
185
186 #endif // LIB_QUENTIER_TYPES_NOTE_H

```

## 6.43 Notebook.h

```

1  /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_NOTEBOOK_H
20 #define LIB_QUENTIER_TYPES_NOTEBOOK_H
21
22 #include "IFavoritableDataElement.h"
23
24 #include <qt5qevercloud/QEverCloud.h>
25
26 #include <QSharedDataPointer>
27
28 namespace quentier {
29
30 QT_FORWARD_DECLARE_CLASS(SharedNotebook)
31 QT_FORWARD_DECLARE_CLASS(User)
32 QT_FORWARD_DECLARE_CLASS(NotebookData)
33
34 class QUINTIER_EXPORT Notebook : public IFavoritableDataElement
35 {
36 public:
37     QN_DECLARE_LOCAL_UID
38     QN_DECLARE_DIRTY
39     QN_DECLARE_LOCAL
40     QN_DECLARE_FAVORITED
41
42 public:
43     explicit Notebook();
44     Notebook(const Notebook & other);
45     Notebook(Notebook && other);
46     Notebook & operator=(const Notebook & other);
47     Notebook & operator=(Notebook && other);
48
49     explicit Notebook(const qevercloud::Notebook & other);
50     explicit Notebook(qevercloud::Notebook && other);
51     Notebook & operator=(const qevercloud::Notebook & other);
52     Notebook & operator=(qevercloud::Notebook && other);
53
54     virtual ~Notebook() override;
55
56     bool operator==(const Notebook & other) const;
57     bool operator!=(const Notebook & other) const;
58
59     const qevercloud::Notebook & qevercloudNotebook() const;
60     qevercloud::Notebook & qevercloudNotebook();
61
62     virtual void clear() override;
63
64     static bool validateName(
65         const QString & name, QString * pErrorDescription = nullptr);
66
67     virtual bool hasGuid() const override;
68     virtual const QString & guid() const override;
69     virtual void setGuid(const QString & guid) override;
70
71     virtual bool hasUpdateSequenceNumber() const override;
72     virtual qint32 updateSequenceNumber() const override;
73     virtual void setUpdateSequenceNumber(const qint32 usn) override;
74
75     virtual bool checkParameters(QString & errorDescription) const override;
76
77     bool hasName() const;
78     const QString & name() const;
79     void setName(const QString & name);
80
81     bool isDefaultNotebook() const;
82     void setDefaultNotebook(const bool defaultNotebook);
83
84     bool hasLinkedNotebookGuid() const;
85     const QString & linkedNotebookGuid() const;

```

```
86     void setLinkedNotebookGuid(const QString & linkedNotebookGuid);
87
88     bool hasCreationTimestamp() const;
89     quint64 creationTimestamp() const;
90     void setCreationTimestamp(const quint64 timestamp);
91
92     bool hasModificationTimestamp() const;
93     quint64 modificationTimestamp() const;
94     void setModificationTimestamp(const quint64 timestamp);
95
96     bool hasPublishingUri() const;
97     const QString & publishingUri() const;
98     void setPublishingUri(const QString & uri);
99
100    bool hasPublishingOrder() const;
101    quint8 publishingOrder() const;
102    void setPublishingOrder(const quint8 order);
103
104    bool hasPublishingAscending() const;
105    bool isPublishingAscending() const;
106    void setPublishingAscending(const bool ascending);
107
108    bool hasPublishingPublicDescription() const;
109    const QString & publishingPublicDescription() const;
110    void setPublishingPublicDescription(
111        const QString & publishingPublicDescription);
112
113    bool hasPublished() const;
114    bool isPublished() const;
115    void setPublished(const bool published);
116
117    bool hasStack() const;
118    const QString & stack() const;
119    void setStack(const QString & stack);
120
121    bool hasSharedNotebooks();
122    QList<SharedNotebook> sharedNotebooks() const;
123    void setSharedNotebooks(QList<qevercloud::SharedNotebook> sharedNotebooks);
124    void setSharedNotebooks(QList<SharedNotebook> && notebooks);
125    void addSharedNotebook(const SharedNotebook & sharedNotebook);
126    void removeSharedNotebook(const SharedNotebook & sharedNotebook);
127
128    bool hasBusinessNotebookDescription() const;
129    const QString & businessNotebookDescription() const;
130
131    void setBusinessNotebookDescription(
132        const QString & businessNotebookDescription);
133
134    bool hasBusinessNotebookPrivilegeLevel() const;
135    quint8 businessNotebookPrivilegeLevel() const;
136    void setBusinessNotebookPrivilegeLevel(const quint8 privilegeLevel);
137
138    bool hasBusinessNotebookRecommended() const;
139    bool isBusinessNotebookRecommended() const;
140    void setBusinessNotebookRecommended(const bool recommended);
141
142    bool hasContact() const;
143    const User contact() const;
144    void setContact(const User & contact);
145
146    bool isLastUsed() const;
147    void setLastUsed(const bool lastUsed);
148
149    bool canReadNotes() const;
150    void setCanReadNotes(const bool canReadNotes);
151
152    bool canCreateNotes() const;
153    void setCanCreateNotes(const bool canCreateNotes);
154
155    bool canUpdateNotes() const;
156    void setCanUpdateNotes(const bool canUpdateNotes);
157
158    bool canExpungeNotes() const;
159    void setCanExpungeNotes(const bool canExpungeNotes);
160
161    bool canShareNotes() const;
162    void setCanShareNotes(const bool canShareNotes);
163
164    bool canEmailNotes() const;
165    void setCanEmailNotes(const bool canEmailNotes);
166
167    bool canSendMessageToRecipients() const;
168    void setCanSendMessageToRecipients(const bool canSendMessageToRecipients);
169
170    bool canUpdateNotebook() const;
171    void setCanUpdateNotebook(const bool canUpdateNotebook);
172
```

```

173     bool canExpungeNotebook() const;
174     void setCanExpungeNotebook(const bool canExpungeNotebook);
175
176     bool canSetDefaultNotebook() const;
177     void setCanSetDefaultNotebook(const bool canSetDefaultNotebook);
178
179     bool canSetNotebookStack() const;
180     void setCanSetNotebookStack(const bool canSetNotebookStack);
181
182     bool canPublishToPublic() const;
183     void setCanPublishToPublic(const bool canPublishToPublic);
184
185     bool canPublishToBusinessLibrary() const;
186     void setCanPublishToBusinessLibrary(const bool canPublishToBusinessLibrary);
187
188     bool canCreateTags() const;
189     void setCanCreateTags(const bool canCreateTags);
190
191     bool canUpdateTags() const;
192     void setCanUpdateTags(const bool canUpdateTags);
193
194     bool canExpungeTags() const;
195     void setCanExpungeTags(const bool canExpungeTags);
196
197     bool canSetParentTag() const;
198     void setCanSetParentTag(const bool canSetParentTag);
199
200     bool canCreateSharedNotebooks() const;
201     void setCanCreateSharedNotebooks(const bool canCreateSharedNotebooks);
202
203     bool canShareNotesWithBusiness() const;
204     void setCanShareNotesWithBusiness(const bool canShareNotesWithBusiness);
205
206     bool canRenameNotebook() const;
207     void setCanRenameNotebook(const bool canRenameNotebook);
208
209     bool hasUpdateWhichSharedNotebookRestrictions() const;
210     qint8 updateWhichSharedNotebookRestrictions() const;
211     void setUpdateWhichSharedNotebookRestrictions(const qint8 which);
212
213     bool hasExpungeWhichSharedNotebookRestrictions() const;
214     qint8 expungeWhichSharedNotebookRestrictions() const;
215     void setExpungeWhichSharedNotebookRestrictions(const qint8 which);
216
217     bool hasRestrictions() const;
218     const qevercloud::NotebookRestrictions & restrictions() const;
219
220     void setNotebookRestrictions(
221         qevercloud::NotebookRestrictions && restrictions);
222
223     bool hasRecipientReminderNotifyEmail() const;
224     bool recipientReminderNotifyEmail() const;
225     void setRecipientReminderNotifyEmail(const bool notifyEmail);
226
227     bool hasRecipientReminderNotifyInApp() const;
228     bool recipientReminderNotifyInApp() const;
229     void setRecipientReminderNotifyInApp(const bool notifyInApp);
230
231     bool hasRecipientInMyList() const;
232     bool recipientInMyList() const;
233     void setRecipientInMyList(const bool inMyList);
234
235     bool hasRecipientStack() const;
236     const QString & recipientStack() const;
237     void setRecipientStack(const QString & recipientString);
238
239     bool hasRecipientSettings() const;
240     const qevercloud::NotebookRecipientSettings & recipientSettings() const;
241     void setNotebookRecipientSettings(
242         qevercloud::NotebookRecipientSettings && settings);
243
244     virtual QTextStream & print(QTextStream & strm) const override;
245
246 private:
247     QSharedDataPointer<NotebookData> d;
248 };
249
250 } // namespace quantier
251
252 Q_DECLARE_METATYPE(quantier::Notebook)
253
254 #endif // LIB_QUENTIER_TYPES_NOTEBOOK_H

```

## 6.44 RegisterMetatypes.h

```

1 /*
2  * Copyright 2016-2019 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_REGISTER_METATYPES_H
20 #define LIB_QUENTIER_TYPES_REGISTER_METATYPES_H
21
22 #include <quentier/utility/Linkage.h>
23
24 namespace quentier {
25
26 void QUENTIER_EXPORT registerMetatypes();
27
28 } // namespace quentier
29
30 #endif // LIB_QUENTIER_TYPES_REGISTER_METATYPES_H

```

## 6.45 Resource.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_RESOURCE_H
20 #define LIB_QUENTIER_TYPES_RESOURCE_H
21
22 #include "INoteStoreDataElement.h"
23 #include "Note.h"
24
25 namespace quentier {
26
27 QT_FORWARD_DECLARE_CLASS(ResourceData)
28
29 class QUENTIER_EXPORT Resource : public INoteStoreDataElement
30 {
31 public:
32     QN_DECLARE_LOCAL_UID
33     QN_DECLARE_DIRTY
34     QN_DECLARE_LOCAL
35
36 public:
37     explicit Resource();
38     Resource(const Resource & other);
39     Resource(Resource && other);
40     explicit Resource(const qevercloud::Resource & resource);
41     Resource & operator=(const Resource & other);
42     Resource & operator=(Resource && other);
43     virtual ~Resource() override;
44
45     bool operator==(const Resource & other) const;
46     bool operator!=(const Resource & other) const;
47
48     const qevercloud::Resource & qevercloudResource() const;

```



```

49     qevercloud::Resource & qevercloudResource();
50
51     virtual void clear() override;
52
53     virtual bool hasGuid() const override;
54     virtual const QString & guid() const override;
55     virtual void setGuid(const QString & guid) override;
56
57     virtual bool hasUpdateSequenceNumber() const override;
58     virtual quint32 updateSequenceNumber() const override;
59
60     virtual void setUpdateSequenceNumber(
61         const quint32 updateSequenceNumber) override;
62
63     virtual bool checkParameters(ErrorString & errorDescription) const override;
64
65     QString displayName() const;
66     void setDisplayName(const QString & displayName);
67
68     QString preferredFileSuffix() const;
69
70     int indexInNote() const;
71     void setIndexInNote(const int index);
72
73     bool hasNoteGuid() const;
74     const QString & noteGuid() const;
75     void setNoteGuid(const QString & guid);
76
77     bool hasNoteLocalUid() const;
78     const QString & noteLocalUid() const;
79     void setNoteLocalUid(const QString & guid);
80
81     bool hasData() const;
82
83     bool hasDataHash() const;
84     const QByteArray & dataHash() const;
85     void setDataHash(const QByteArray & hash);
86
87     bool hasDataSize() const;
88     quint32 dataSize() const;
89     void setDataSize(const quint32 size);
90
91     bool hasDataBody() const;
92     const QByteArray & dataBody() const;
93     void setDataBody(const QByteArray & body);
94
95     bool hasMime() const;
96     const QString & mime() const;
97     void setMime(const QString & mime);
98
99     bool hasWidth() const;
100    quint16 width() const;
101    void setWidth(const quint16 width);
102
103    bool hasHeight() const;
104    quint16 height() const;
105    void setHeight(const quint16 height);
106
107    bool hasRecognitionData() const;
108
109    bool hasRecognitionDataHash() const;
110    const QByteArray & recognitionDataHash() const;
111    void setRecognitionDataHash(const QByteArray & hash);
112
113    bool hasRecognitionDataSize() const;
114    quint32 recognitionDataSize() const;
115    void setRecognitionDataSize(const quint32 size);
116
117    bool hasRecognitionDataBody() const;
118    const QByteArray & recognitionDataBody() const;
119    void setRecognitionDataBody(const QByteArray & body);
120
121    bool hasAlternateData() const;
122
123    bool hasAlternateDataHash() const;
124    const QByteArray & alternateDataHash() const;
125    void setAlternateDataHash(const QByteArray & hash);
126
127    bool hasAlternateDataSize() const;
128    quint32 alternateDataSize() const;
129    void setAlternateDataSize(const quint32 size);
130
131    bool hasAlternateDataBody() const;
132    const QByteArray & alternateDataBody() const;
133    void setAlternateDataBody(const QByteArray & body);
134
135    bool hasResourceAttributes() const;

```

```

136     const qevercloud::ResourceAttributes & resourceAttributes() const;
137     qevercloud::ResourceAttributes & resourceAttributes();
138
139     void setResourceAttributes(
140         const qevercloud::ResourceAttributes & attributes);
141
142     void setResourceAttributes(qevercloud::ResourceAttributes && attributes);
143
144     friend class Note;
145
146     virtual QTextStream & print(QTextStream & strm) const override;
147
148 private:
149     QSharedDataPointer<ResourceData> d;
150 };
151
152 } // namespace quantier
153
154 #endif // LIB_QUENTIER_TYPES_RESOURCE_H

```

## 6.46 ResourceRecognitionIndexItem.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_RESOURCE_RECOGNITION_INDEX_ITEM_H
20 #define LIB_QUENTIER_TYPES_RESOURCE_RECOGNITION_INDEX_ITEM_H
21
22 #include <quantier/utility/Linkage.h>
23 #include <quantier/utility/Printable.h>
24
25 #include <QByteArray>
26 #include <QSharedDataPointer>
27
28 namespace quantier {
29
30 QT_FORWARD_DECLARE_CLASS(ResourceRecognitionIndexItemData)
31
32 class QUENTIER_EXPORT ResourceRecognitionIndexItem : public Printable
33 {
34 public:
35     explicit ResourceRecognitionIndexItem();
36     ResourceRecognitionIndexItem(const ResourceRecognitionIndexItem & other);
37
38     ResourceRecognitionIndexItem & operator=(
39         const ResourceRecognitionIndexItem & other);
40
41     virtual ~ResourceRecognitionIndexItem() override;
42
43     bool isValid() const;
44
45     int x() const;
46     void setX(const int x);
47
48     int y() const;
49     void setY(const int y);
50
51     int h() const;
52     void setH(const int h);
53
54     int w() const;
55     void setW(const int w);
56
57     int offset() const;
58     void setOffset(const int offset);
59
60     int duration() const;
61     void setDuration(const int duration);

```

```

62
63     QVector<int> strokeList() const;
64     int numStrokes() const;
65     bool strokeAt(const int strokeIndex, int & stroke) const;
66     bool setStrokeAt(const int strokeIndex, const int stroke);
67     void setStrokeList(const QVector<int> & strokeList);
68     void reserveStrokeListSpace(const int numItems);
69     void addStroke(const int stroke);
70     bool removeStroke(const int stroke);
71     bool removeStrokeAt(const int strokeIndex);
72
73     struct TextItem
74     {
75         bool operator==(const TextItem & other) const
76     {
77         return (m_text == other.m_text) && (m_weight == other.m_weight);
78     }
79
80         QString m_text;
81         int m_weight = -1;
82     };
83
84     QVector<TextItem> textItems() const;
85     int numTextItems() const;
86     bool textItemAt(const int textItemIndex, TextItem & textItem) const;
87     bool setTextItemAt(const int textItemIndex, const TextItem & textItem);
88     void setTextItems(const QVector<TextItem> & textItems);
89     void reserveTextItemsSpace(const int numItems);
90     void addTextItem(const TextItem & item);
91     bool removeTextItem(const TextItem & item);
92     bool removeTextItemAt(const int textItemIndex);
93
94     struct ObjectItem
95     {
96         bool operator==(const ObjectItem & other) const
97     {
98         return (m_objectType == other.m_objectType) &&
99             (m_weight == other.m_weight);
100     }
101
102         QString m_objectType;
103         int m_weight = -1;
104     };
105
106     QVector<ObjectItem> objectItems() const;
107     int numObjectItems() const;
108     bool objectItemAt(const int objectItemIndex, ObjectItem & objectItem) const;
109
110     bool setObjectItemAt(
111         const int objectItemIndex, const ObjectItem & objectItem);
112
113     void setObjectItems(const QVector<ObjectItem> & objectItems);
114     void reserveObjectItemsSpace(const int numItems);
115     void addObjectItem(const ObjectItem & item);
116     bool removeObjectItem(const ObjectItem & item);
117     bool removeObjectItemAt(const int objectItemIndex);
118
119     struct ShapeItem
120     {
121         bool operator==(const ShapeItem & other) const
122     {
123         return (m_shapeType == other.m_shapeType) &&
124             (m_weight == other.m_weight);
125     }
126
127         QString m_shapeType;
128         int m_weight = -1;
129     };
130
131     QVector<ShapeItem> shapeItems() const;
132     int numShapeItems() const;
133     bool shapeItemAt(const int shapeItemIndex, ShapeItem & shapeItem) const;
134     bool setShapeItemAt(const int shapeItemIndex, const ShapeItem & shapeItem);
135     void setShapeItems(const QVector<ShapeItem> & shapeItems);
136     void reserveShapeItemsSpace(const int numItems);
137     void addShapeItem(const ShapeItem & item);
138     bool removeShapeItem(const ShapeItem & item);
139     bool removeShapeItemAt(const int shapeItemIndex);
140
141     struct BarcodeItem
142     {
143         bool operator==(const BarcodeItem & other) const
144     {
145         return (m_barcode == other.m_barcode) &&
146             (m_weight == other.m_weight);
147     }
148

```

```

149     QString m_barcode;
150     int m_weight = -1;
151 };
152
153 QVector<BarcodeItem> barcodeItems() const;
154 int numBarcodeItems() const;
155
156 bool barcodeItemAt(
157     const int barcodeItemIndex, BarcodeItem & barcodeItem) const;
158
159 bool setBarcodeItemAt(
160     const int barcodeItemIndex, const BarcodeItem & barcodeItem);
161
162 void setBarcodeItems(const QVector<BarcodeItem> & barcodeItems);
163 void reserveBarcodeItemsSpace(const int numItems);
164 void addBarcodeItem(const BarcodeItem & item);
165 bool removeBarcodeItem(const BarcodeItem & item);
166 bool removeBarcodeItemAt(const int barcodeItemIndex);
167
168 virtual QTextStream & print(QTextStream & strm) const override;
169
170 private:
171     QSharedDataPointer<ResourceRecognitionIndexItemData> d;
172 };
173
174 } // namespace quantier
175
176 #endif // LIB_QUENTIER_TYPES_RESOURCE_RECOGNITION_INDEX_ITEM_H

```

## 6.47 ResourceRecognitionIndices.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquantier
5  *
6  * libquantier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquantier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquantier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_RESOURCE_RECOGNITION_INDICES_H
20 #define LIB_QUENTIER_TYPES_RESOURCE_RECOGNITION_INDICES_H
21
22 #include <quantier/types/ResourceRecognitionIndexItem.h>
23
24 #include <QByteArray>
25 #include <QSharedDataPointer>
26 #include <QVector>
27
28 namespace quantier {
29
30 QT_FORWARD_DECLARE_CLASS(ResourceRecognitionIndicesData)
31
32 class QUENTIER_EXPORT ResourceRecognitionIndices : public Printable
33 {
34 public:
35     explicit ResourceRecognitionIndices();
36     ResourceRecognitionIndices(const ResourceRecognitionIndices & other);
37
38     explicit ResourceRecognitionIndices(
39         const QByteArray & rawRecognitionIndicesData);
40
41     ResourceRecognitionIndices & operator=(
42         const ResourceRecognitionIndices & other);
43
44     virtual ~ResourceRecognitionIndices() override;
45
46     bool isNull() const;
47     bool isValid() const;
48
49     QString objectId() const;
50     QString objectType() const;
51     QString recoType() const;
52     QString engineVersion() const;

```

```

53     QString docType() const;
54     QString lang() const;
55
56     int objectHeight() const;
57     int objectWidth() const;
58
59     QVector<ResourceRecognitionIndexItem> items() const;
60
61     bool setData(const QByteArray & rawRecognitionIndicesData);
62
63     virtual QTextStream & print(QTextStream & strm) const override;
64
65 private:
66     QSharedDataPointer<ResourceRecognitionIndicesData> d;
67 };
68
69 } // namespace quentier
70
71 #endif // LIB_QUENTIER_TYPES_RESOURCE_RECOGNITION_INDICES_H

```

## 6.48 SavedSearch.h

```

1  /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_SAVED_SEARCH_H
20 #define LIB_QUENTIER_TYPES_SAVED_SEARCH_H
21
22 #include "IFavoritableDataElement.h"
23
24 #include <qt5qevercloud/QEverCloud.h>
25
26 #include <QSharedDataPointer>
27
28 namespace quentier {
29
30 QT_FORWARD_DECLARE_CLASS(SavedSearchData)
31
32 class QUENTIER_EXPORT SavedSearch : public IFavoritableDataElement
33 {
34 public:
35     QN_DECLARE_LOCAL_UID
36     QN_DECLARE_DIRTY
37     QN_DECLARE_LOCAL
38     QN_DECLARE_FAVORITED
39
40 public:
41     using QueryFormat = qevercloud::QueryFormat;
42     using SavedSearchScope = qevercloud::SavedSearchScope;
43
44 public:
45     explicit SavedSearch();
46     SavedSearch(const SavedSearch & other);
47     SavedSearch(SavedSearch && other);
48     SavedSearch & operator=(const SavedSearch & other);
49     SavedSearch & operator=(SavedSearch && other);
50
51     explicit SavedSearch(const qevercloud::SavedSearch & search);
52     explicit SavedSearch(qevercloud::SavedSearch && search);
53
54     virtual ~SavedSearch() override;
55
56     const qevercloud::SavedSearch & qevercloudSavedSearch() const;
57     qevercloud::SavedSearch & qevercloudSavedSearch();
58
59     bool operator==(const SavedSearch & other) const;
60     bool operator!=(const SavedSearch & other) const;
61

```

```

62     virtual void clear() override;
63
64     static bool validateName(
65         const QString & name, QString * pErrorDescription = nullptr);
66
67     virtual bool hasGuid() const override;
68     virtual const QString & guid() const override;
69     virtual void setGuid(const QString & guid) override;
70
71     virtual bool hasUpdateSequenceNumber() const override;
72     virtual quint32 updateSequenceNumber() const override;
73     virtual void setUpdateSequenceNumber(const quint32 usn) override;
74
75     virtual bool checkParameters(QString & errorDescription) const override;
76
77     bool hasName() const;
78     const QString & name() const;
79     void setName(const QString & name);
80
81     bool hasQuery() const;
82     const QString & query() const;
83     void setQuery(const QString & query);
84
85     bool hasQueryFormat() const;
86     QueryFormat queryFormat() const;
87     void setQueryFormat(const quint8 queryFormat);
88
89     bool hasIncludeAccount() const;
90     bool includeAccount() const;
91     void setIncludeAccount(const bool includeAccount);
92
93     bool hasIncludePersonalLinkedNotebooks() const;
94     bool includePersonalLinkedNotebooks() const;
95
96     void setIncludePersonalLinkedNotebooks(
97         const bool includePersonalLinkedNotebooks);
98
99     bool hasIncludeBusinessLinkedNotebooks() const;
100    bool includeBusinessLinkedNotebooks() const;
101
102    void setIncludeBusinessLinkedNotebooks(
103        const bool includeBusinessLinkedNotebooks);
104
105    virtual QTextStream & print(QTextStream & strm) const override;
106
107 private:
108     QSharedDataPointer<SavedSearchData> d;
109 };
110
111 } // namespace quentier
112
113 Q_DECLARE_METATYPE(quentier::SavedSearch)
114
115 #endif // LIB_QUENTIER_TYPES_SAVED_SEARCH_H

```

## 6.49 SharedNote.h

```

1  /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_SHARED_NOTE_H
20 #define LIB_QUENTIER_TYPES_SHARED_NOTE_H
21
22 #include <quentier/utility/Linkage.h>
23 #include <quentier/utility/Printable.h>
24
25 #include <qt5qevercloud/QEverCloud.h>
26

```

```

27 #include <QSharedDataPointer>
28
29 namespace quantier {
30
31 QT_FORWARD_DECLARE_CLASS(SharedNoteData)
32
33 class QUANTIER_EXPORT SharedNote : public Printable
34 {
35 public:
36     using SharedNotePrivilegeLevel = qevercloud::SharedNotePrivilegeLevel;
37     using ContactType = qevercloud::ContactType;
38
39 public:
40     explicit SharedNote();
41     SharedNote(const SharedNote & other);
42     SharedNote(SharedNote && other);
43     explicit SharedNote(const qevercloud::SharedNote & sharedNote);
44     SharedNote & operator=(const SharedNote & other);
45     SharedNote & operator=(SharedNote && other);
46     virtual ~SharedNote() override;
47
48     bool operator==(const SharedNote & other) const;
49     bool operator!=(const SharedNote & other) const;
50
51     const qevercloud::SharedNote & qevercloudSharedNote() const;
52     qevercloud::SharedNote & qevercloudSharedNote();
53
54     const QString & noteGuid() const;
55     void setNoteGuid(const QString & noteGuid);
56
57     int indexInNote() const;
58     void setIndexInNote(const int index);
59
60     bool hasSharerUserId() const;
61     quint32 sharerUserId() const;
62     void setSharerUserId(const quint32 id);
63
64     bool hasRecipientIdentityId() const;
65     quint64 recipientIdentityId() const;
66     void setRecipientIdentityId(const quint64 identityId);
67
68     bool hasRecipientIdentityContactName() const;
69     const QString & recipientIdentityContactName() const;
70
71     void setRecipientIdentityContactName(
72         const QString & recipientIdentityContactName);
73
74     bool hasRecipientIdentityContactId() const;
75     const QString & recipientIdentityContactId() const;
76
77     void setRecipientIdentityContactId(
78         const QString & recipientIdentityContactId);
79
80     bool hasRecipientIdentityContactType() const;
81     ContactType recipientIdentityContactType() const;
82
83     void setRecipientIdentityContactType(
84         const ContactType recipientIdentityContactType);
85
86     void setRecipientIdentityContactType(
87         const quint32 recipientIdentityContactType);
88
89     bool hasRecipientIdentityContactPhotoUrl() const;
90     const QString & recipientIdentityContactPhotoUrl() const;
91
92     void setRecipientIdentityContactPhotoUrl(
93         const QString & recipientIdentityPhotoUrl);
94
95     bool hasRecipientIdentityContactPhotoLastUpdated() const;
96     quint64 recipientIdentityContactPhotoLastUpdated() const;
97
98     void setRecipientIdentityContactPhotoLastUpdated(
99         const quint64 recipientIdentityPhotoLastUpdated);
100
101     bool hasRecipientIdentityContactMessagingPermit() const;
102     const QByteArray & recipientIdentityContactMessagingPermit() const;
103
104     void setRecipientIdentityContactMessagingPermit(
105         const QByteArray & messagingPermit);
106
107     bool hasRecipientIdentityContactMessagingPermitExpires() const;
108     quint64 recipientIdentityContactMessagingPermitExpires() const;
109
110     void setRecipientIdentityContactMessagingPermitExpires(
111         const quint64 timestamp);
112
113     bool hasRecipientIdentityUserId() const;

```

```

114     quint32 recipientIdentityUserId() const;
115     void setRecipientIdentityUserId(const quint32 userId);
116
117     bool hasRecipientIdentityDeactivated() const;
118     bool recipientIdentityDeactivated() const;
119     void setRecipientIdentityDeactivated(const bool deactivated);
120
121     bool hasRecipientIdentitySameBusiness() const;
122     bool recipientIdentitySameBusiness() const;
123     void setRecipientIdentitySameBusiness(const bool sameBusiness);
124
125     bool hasRecipientIdentityBlocked() const;
126     bool recipientIdentityBlocked() const;
127     void setRecipientIdentityBlocked(const bool blocked);
128
129     bool hasRecipientIdentityUserConnected() const;
130     bool recipientIdentityUserConnected() const;
131     void setRecipientIdentityUserConnected(const bool userConnected);
132
133     bool hasRecipientIdentityEventId() const;
134     quint64 recipientIdentityEventId() const;
135     void setRecipientIdentityEventId(const quint64 eventId);
136
137     bool hasRecipientIdentity() const;
138     const qevercloud::Identity & recipientIdentity() const;
139     void setRecipientIdentity(qevercloud::Identity && identity);
140
141     bool hasPrivilegeLevel() const;
142     SharedNotePrivilegeLevel privilegeLevel() const;
143     void setPrivilegeLevel(const SharedNotePrivilegeLevel level);
144     void setPrivilegeLevel(const quint8 level);
145
146     bool hasCreationTimestamp() const;
147     quint64 creationTimestamp() const;
148     void setCreationTimestamp(const quint64 timestamp);
149
150     bool hasModificationTimestamp() const;
151     quint64 modificationTimestamp() const;
152     void setModificationTimestamp(const quint64 timestamp);
153
154     bool hasAssignmentTimestamp() const;
155     quint64 assignmentTimestamp() const;
156     void setAssignmentTimestamp(const quint64 timestamp);
157
158     virtual QTextStream & print(QTextStream & strm) const override;
159
160 private:
161     QSharedDataPointer<SharedNoteData> d;
162 };
163
164 } // namespace quentier
165
166 #endif // LIB_QUENTIER_TYPES_SHARED_NOTE_H

```

## 6.50 SharedNotebook.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_SHARED_NOTEBOOK_H
20 #define LIB_QUENTIER_TYPES_SHARED_NOTEBOOK_H
21
22 #include <quentier/utility/Printable.h>
23
24 #include <qt5qevercloud/QEverCloud.h>
25
26 #include <QSharedDataPointer>
27

```



```

28 namespace quantier {
29
30 QT_FORWARD_DECLARE_CLASS(SharedNotebookData)
31
32 class QUANTIER_EXPORT SharedNotebook : public Printable
33 {
34 public:
35     using SharedNotebookPrivilegeLevel =
36         qevercloud::SharedNotebookPrivilegeLevel;
37
38 public:
39     explicit SharedNotebook();
40     SharedNotebook(const SharedNotebook & other);
41     SharedNotebook(SharedNotebook && other);
42
43     explicit SharedNotebook(
44         const qevercloud::SharedNotebook & qecSharedNotebook);
45
46     SharedNotebook & operator=(const SharedNotebook & other);
47     SharedNotebook & operator=(SharedNotebook && other);
48
49     virtual ~SharedNotebook() override;
50
51     bool operator==(const SharedNotebook & other) const;
52     bool operator!=(const SharedNotebook & other) const;
53
54     const qevercloud::SharedNotebook & qevercloudSharedNotebook() const;
55     qevercloud::SharedNotebook & qevercloudSharedNotebook();
56
57     int indexInNotebook() const;
58     void setIndexInNotebook(const int index);
59
60     bool hasId() const;
61     quint64 id() const;
62     void setId(const quint64 id);
63
64     bool hasUserId() const;
65     quint32 userId() const;
66     void setUserId(const quint32 userId);
67
68     bool hasNotebookGuid() const;
69     const QString & notebookGuid() const;
70     void setNotebookGuid(const QString & notebookGuid);
71
72     bool hasEmail() const;
73     const QString & email() const;
74     void setEmail(const QString & email);
75
76     bool hasCreationTimestamp() const;
77     quint64 creationTimestamp() const;
78     void setCreationTimestamp(const quint64 timestamp);
79
80     bool hasModificationTimestamp() const;
81     quint64 modificationTimestamp() const;
82     void setModificationTimestamp(const quint64 timestamp);
83
84     bool hasUsername() const;
85     const QString & username() const;
86     void setUsername(const QString & username);
87
88     bool hasPrivilegeLevel() const;
89     SharedNotebookPrivilegeLevel privilegeLevel() const;
90     void setPrivilegeLevel(const SharedNotebookPrivilegeLevel privilegeLevel);
91     void setPrivilegeLevel(const quint8 privilegeLevel);
92
93     bool hasReminderNotifyEmail() const;
94     bool reminderNotifyEmail() const;
95     void setReminderNotifyEmail(const bool notifyEmail);
96
97     bool hasReminderNotifyApp() const;
98     bool reminderNotifyApp() const;
99     void setReminderNotifyApp(const bool notifyApp);
100
101     bool hasRecipientUsername() const;
102     const QString & recipientUsername() const;
103     void setRecipientUsername(const QString & recipientUsername);
104
105     bool hasRecipientUserId() const;
106     quint32 recipientUserId() const;
107     void setRecipientUserId(const quint32 userId);
108
109     bool hasRecipientIdentityId() const;
110     quint64 recipientIdentityId() const;
111     void setRecipientIdentityId(const quint64 recipientIdentityId);
112
113     bool hasGlobalId() const;
114     const QString & globalId() const;

```

```

115     void setGlobalId(const QString & globalId);
116
117     bool hasSharerUserId() const;
118     quint32 sharerUserId() const;
119     void setSharerUserId(quint32 sharerUserId);
120
121     bool hasAssignmentTimestamp() const;
122     quint64 assignmentTimestamp() const;
123     void setAssignmentTimestamp(const quint64 timestamp);
124
125     virtual QTextStream & print(QTextStream & strm) const override;
126
127     friend class Notebook;
128
129 private:
130     QSharedDataPointer<SharedNotebookData> d;
131 };
132
133 } // namespace quentier
134
135 #endif // LIB_QUENTIER_TYPES_SHARED_NOTEBOOK_H

```

## 6.51 Tag.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_TAG_H
20 #define LIB_QUENTIER_TYPES_TAG_H
21
22 #include "IFavoritableDataElement.h"
23
24 #include <qt5qevercloud/QEverCloud.h>
25
26 #include <QSharedDataPointer>
27
28 namespace quentier {
29
30 QT_FORWARD_DECLARE_CLASS(TagData)
31
32 class QUENTIER_EXPORT Tag : public IFavoritableDataElement
33 {
34 public:
35     QN_DECLARE_LOCAL_UID
36     QN_DECLARE_DIRTY
37     QN_DECLARE_LOCAL
38     QN_DECLARE_FAVORITED
39
40 public:
41     explicit Tag();
42     Tag(const Tag & other);
43     Tag(Tag && other);
44     Tag & operator=(const Tag & other);
45     Tag & operator=(Tag && other);
46
47     explicit Tag(const qevercloud::Tag & other);
48     explicit Tag(qevercloud::Tag && other);
49
50     virtual ~Tag() override;
51
52     bool operator==(const Tag & other) const;
53     bool operator!=(const Tag & other) const;
54
55     const qevercloud::Tag & qevercloudTag() const;
56     qevercloud::Tag & qevercloudTag();
57
58     virtual void clear() override;
59

```

```

60     static bool validateName(
61         const QString & name, ErrorString * pErrorDescription = nullptr);
62
63     virtual bool hasGuid() const override;
64     virtual const QString & guid() const override;
65     virtual void setGuid(const QString & guid) override;
66
67     virtual bool hasUpdateSequenceNumber() const override;
68     virtual qint32 updateSequenceNumber() const override;
69     virtual void setUpdateSequenceNumber(const qint32 usn) override;
70
71     virtual bool checkParameters(ErrorString & errorDescription) const override;
72
73     bool hasName() const;
74     const QString & name() const;
75     void setName(const QString & name);
76
77     bool hasParentGuid() const;
78     const QString & parentGuid() const;
79     void setParentGuid(const QString & parentGuid);
80
81     bool hasParentLocalUid() const;
82     const QString & parentLocalUid() const;
83     void setParentLocalUid(const QString & parentLocalUid);
84
85     bool hasLinkedNotebookGuid() const;
86     const QString & linkedNotebookGuid() const;
87     void setLinkedNotebookGuid(const QString & linkedNotebookGuid);
88
89     virtual QTextStream & print(QTextStream & strm) const override;
90
91 private:
92     QSharedDataPointer<TagData> d;
93 };
94
95 } // namespace quentier
96
97 Q_DECLARE_METATYPE(quentier::Tag)
98
99 #endif // LIB_QUENTIER_TYPES_TAG_H

```

## 6.52 User.h

```

1  /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_TYPES_USER_H
20 #define LIB_QUENTIER_TYPES_USER_H
21
22 #include <quentier/types/ErrorMessage.h>
23 #include <quentier/utility/Printable.h>
24
25 #include <qt5qevercloud/QEverCloud.h>
26
27 #include <QtGlobal>
28
29 namespace quentier {
30
31 QT_FORWARD_DECLARE_CLASS(UserData)
32
33 class QUENTIER_EXPORT User : public Printable
34 {
35 public:
36     using PrivilegeLevel = qevercloud::PrivilegeLevel;
37     using ServiceLevel = qevercloud::ServiceLevel;
38
39 public:
40     explicit User();

```

```

41     explicit User(const qevercloud::User & user);
42     explicit User(qevercloud::User && user);
43     User(const User & other);
44     User(User && other);
45     User & operator=(const User & other);
46     User & operator=(User && other);
47     virtual ~User() override;
48
49     bool operator==(const User & other) const;
50     bool operator!=(const User & other) const;
51
52     const qevercloud::User & qevercloudUser() const;
53     qevercloud::User & qevercloudUser();
54
55     void clear();
56
57     bool isDirty() const;
58     void setDirty(const bool dirty);
59
60     bool isLocal() const;
61     void setLocal(const bool local);
62
63     bool checkParameters(ErrorString & errorDescription) const;
64
65     bool hasId() const;
66     qint32 id() const;
67     void setId(const qint32 id);
68
69     bool hasUsername() const;
70     const QString & username() const;
71     void setUsername(const QString & username);
72
73     bool hasEmail() const;
74     const QString & email() const;
75     void setEmail(const QString & email);
76
77     bool hasName() const;
78     const QString & name() const;
79     void setName(const QString & name);
80
81     bool hasTimezone() const;
82     const QString & timezone() const;
83     void setTimezone(const QString & timezone);
84
85     bool hasPrivilegeLevel() const;
86     PrivilegeLevel privilegeLevel() const;
87     void setPrivilegeLevel(const qint8 level);
88
89     bool hasServiceLevel() const;
90     ServiceLevel serviceLevel() const;
91     void setServiceLevel(const qint8 level);
92
93     bool hasCreationTimestamp() const;
94     qint64 creationTimestamp() const;
95     void setCreationTimestamp(const qint64 timestamp);
96
97     bool hasModificationTimestamp() const;
98     qint64 modificationTimestamp() const;
99     void setModificationTimestamp(const qint64 timestamp);
100
101     bool hasDeletionTimestamp() const;
102     qint64 deletionTimestamp() const;
103     void setDeletionTimestamp(const qint64 timestamp);
104
105     bool hasActive() const;
106     bool active() const;
107     void setActive(const bool active);
108
109     bool hasShardId() const;
110     const QString & shardId() const;
111     void setShardId(const QString & shardId);
112
113     bool hasUserAttributes() const;
114     const qevercloud::UserAttributes & userAttributes() const;
115     void setUserAttributes(qevercloud::UserAttributes && attributes);
116
117     bool hasAccounting() const;
118     const qevercloud::Accounting & accounting() const;
119     void setAccounting(qevercloud::Accounting && accounting);
120
121     bool hasBusinessUserInfo() const;
122     const qevercloud::BusinessUserInfo & businessUserInfo() const;
123     void setBusinessUserInfo(qevercloud::BusinessUserInfo && info);
124
125     bool hasPhotoUrl() const;
126     QString photoUrl() const;
127     void setPhotoUrl(const QString & photoUrl);

```

```

128
129     bool hasPhotoLastUpdateTimestamp() const;
130     qint64 photoLastUpdateTimestamp() const;
131     void setPhotoLastUpdateTimestamp(const qint64 timestamp);
132
133     bool hasAccountLimits() const;
134     const qevercloud::AccountLimits & accountLimits() const;
135     void setAccountLimits(qevercloud::AccountLimits && limits);
136
137     virtual QTextStream & print(QTextStream & strm) const override;
138
139     friend class Notebook;
140
141 private:
142     QSharedDataPointer<UserData> d;
143 };
144
145 } // namespace quentier
146
147 #endif // LIB_QUENTIER_TYPES_USER_H

```

## 6.53 ApplicationSettings.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_APPLICATION_SETTINGS_H
20 #define LIB_QUENTIER_UTILITY_APPLICATION_SETTINGS_H
21
22 #include <quentier/types/Account.h>
23
24 #include <QSettings>
25
26 namespace quentier {
27
28     class QUENTIER_EXPORT ApplicationSettings : public QSettings, public Printable
29     {
30     public:
31         Q_OBJECT
32     public:
33         ApplicationSettings(const QString & settingsName = {});
34
35         ApplicationSettings(
36             const Account & account, const QString & settingsName = {});
37
38         ApplicationSettings(
39             const Account & account, const char * settingsName,
40             const int settingsNameSize = -1);
41
42         virtual ~ApplicationSettings() override;
43
44     public:
45         struct ArrayCloser
46         {
47             ArrayCloser(ApplicationSettings & settings) : m_settings(settings) {}
48
49             ~ArrayCloser()
50             {
51                 m_settings.endArray();
52                 m_settings.sync();
53             }
54
55             ApplicationSettings & m_settings;
56         };
57
58         struct GroupCloser
59         {
60             GroupCloser(ApplicationSettings & settings) : m_settings(settings) {}
61
62             ~GroupCloser()
63             {
64                 m_settings.endGroup();
65                 m_settings.sync();
66             }
67
68             ApplicationSettings & m_settings;
69         };
70
71     private:
72         void sync();
73
74     private:
75         void endArray();
76         void endGroup();
77         void sync();
78     };
79
80 }
81
82 #endif

```

```

113     ~GroupCloser()
114     {
115         m_settings.endGroup();
116         m_settings.sync();
117     }
118
119     ApplicationSettings & m_settings;
120 };
121
122 public:
123     void beginGroup(const QString & prefix);
124
125     void beginGroup(const char * prefix, const int size = -1);
126
127     int beginReadArray(const QString & prefix);
128
129     int beginReadArray(const char * prefix, const int size = -1);
130
131     void beginWriteArray(const QString & prefix, const int arraySize = -1);
132
133     void beginWriteArray(
134         const char * prefix, const int arraySize = -1,
135         const int prefixSize = -1);
136
137     bool contains(const QString & key) const;
138
139     bool contains(const char * key, const int size = -1) const;
140
141     void remove(const QString & key);
142
143     void remove(const char * key, const int size = -1);
144
145     void setValue(const QString & key, const QVariant & value);
146
147     void setValue(
148         const char * key, const QVariant & value, const int keySize = -1);
149
150     QVariant value(
151         const QString & key, const QVariant & defaultValue = {}) const;
152
153     QVariant value(
154         const char * key, const QVariant & defaultValue = {},
155         const int keySize = -1) const;
156
157 public:
158     virtual QTextStream & print(QTextStream & strm) const override;
159
160 private:
161     Q_DISABLE_COPY(ApplicationSettings)
162 };
163
164 } // namespace quentier
165
166 #endif // LIB_QUENTIER_UTILITY_APPLICATION_SETTINGS_H

```

## 6.54 Checks.h

```

1 /*
2  * Copyright 2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_CHECKS_H
20 #define LIB_QUENTIER_UTILITY_CHECKS_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QString>
25
26 namespace quentier {

```

```

27
34 bool QUENTIER_EXPORT checkGuid(const QString & guid);
35
45 bool QUENTIER_EXPORT
46 checkUpdateSequenceNumber(const int32_t updateSequenceNumber);
47
48 } // namespace quentier
49
50 #endif // LIB_QUENTIER_UTILITY_CHECKS_H

```

## 6.55 Compat.h

```

1 /*
2  * Copyright 2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_COMPAT_H
20 #define LIB_QUENTIER_UTILITY_COMPAT_H
21
22 #include <QHash>
23 #include <QString>
24 #include <QtGlobal>
25
26 #if QT_VERSION < QT_VERSION_CHECK(5, 7, 0)
27 #include <type_traits>
28 #endif
29
30 // Compatibility with older Qt versions
31
32 #if QT_VERSION < QT_VERSION_CHECK(5, 7, 0)
33
34 // this adds const to non-const objects (like std::as_const)
35 template <typename T>
36 Q_DECL_CONSTEXPR typename std::add_const<T>::type & qAsConst(T & t)
37     Q_DECL_NOTHROW
38 {
39     return t;
40 }
41
42 // prevent rvalue arguments:
43 template <typename T>
44 void qAsConst(const T &&) = delete;
45
46 #endif
47
48 // Compatibility with boost parts which require to take a hash of QString
49
50 inline std::size_t hash_value(QString x) noexcept
51 {
52     return qHash(x);
53 }
54
55 #endif // LIB_QUENTIER_UTILITY_COMPAT_H

```

## 6.56 DateTime.h

```

1 /*
2  * Copyright 2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *

```

```

10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_DATE_TIME_H
20 #define LIB_QUENTIER_UTILITY_DATE_TIME_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QFlags>
25
26 namespace quentier {
27
28 constexpr int secondsToMilliseconds(int seconds) noexcept
29 {
30     return seconds * 1000;
31 }
32
33 class QUENTIER_EXPORT DateTimePrint
34 {
35 public:
36     enum Option
37     {
38         IncludeNumericTimestamp = 1 « 1,
39         IncludeMilliseconds = 1 « 2,
40         IncludeTimezone = 1 « 3
41     };
42     Q_DECLARE_FLAGS(Options, Option)
43 };
44
45 Q_DECLARE_OPERATORS_FOR_FLAGS(DateTimePrint::Options)
46
47 const QString QUENTIER_EXPORT printableDateTimeFromTimestamp(
48     const qint64 timestamp,
49     DateTimePrint::Options options = DateTimePrint::Options(
50         DateTimePrint::IncludeNumericTimestamp |
51         DateTimePrint::IncludeMilliseconds | DateTimePrint::IncludeTimezone),
52     const char * customFormat = nullptr);
53
54 } // namespace quentier
55
56 #endif // LIB_QUENTIER_UTILITY_DATE_TIME_H

```

## 6.57 EncryptionManager.h

```

1 /*
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_ENCRYPTION_MANAGER_H
20 #define LIB_QUENTIER_UTILITY_ENCRYPTION_MANAGER_H
21
22 #include <quentier/types/ErrorMessage.h>
23 #include <quentier/utility/Linkage.h>
24
25 #include <QObject>
26 #include <QString>
27 #include <QUuid>
28
29 namespace quentier {
30
31 QT_FORWARD_DECLARE_CLASS(EncryptionManagerPrivate)
32

```



```

33
34 class QUENTIER_EXPORT EncryptionManager : public QObject
35 {
36     Q_OBJECT
37 public:
38     explicit EncryptionManager(QObject * parent = nullptr);
39     virtual ~EncryptionManager();
40
41     bool decrypt(
42         const QString & encryptedText, const QString & passphrase,
43         const QString & cipher, const size_t keyLength, QString & decryptedText,
44         ErrorString & errorDescription);
45
46     bool encrypt(
47         const QString & textToEncrypt, const QString & passphrase,
48         QString & cipher, size_t & keyLength, QString & encryptedText,
49         ErrorString & errorDescription);
50
51     Q_SIGNALS:
52     void decryptedText(
53         QString text, bool success, ErrorString errorDescription,
54         QUuid requestId);
55
56     void encryptedText(
57         QString encryptedText, bool success, ErrorString errorDescription,
58         QUuid requestId);
59
60 public Q_SLOTS:
61     void onDecryptTextRequest(
62         QString encryptedText, QString passphrase, QString cipher,
63         size_t keyLength, QUuid requestId);
64
65     void onEncryptTextRequest(
66         QString textToEncrypt, QString passphrase, QString cipher,
67         size_t keyLength, QUuid requestId);
68
69 private:
70     EncryptionManagerPrivate * const d_ptr;
71     Q_DECLARE_PRIVATE(EncryptionManager)
72 };
73
74 } // namespace quentier
75
76 #endif // LIB_QUENTIER_UTILITY_ENCRYPTION_MANAGER_H

```

## 6.58 EventLoopWithExitStatus.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_EVENT_LOOP_WITH_EXIT_STATUS_H
20 #define LIB_QUENTIER_UTILITY_EVENT_LOOP_WITH_EXIT_STATUS_H
21
22 #include <quentier/types/ErrorString.h>
23 #include <quentier/utility/Linkage.h>
24
25 #include <QEventLoop>
26
27 QT_FORWARD_DECLARE_CLASS(QDebug)
28 QT_FORWARD_DECLARE_CLASS(QTextStream)
29
30 namespace quentier {
31
32 class QUENTIER_EXPORT EventLoopWithExitStatus : public QEventLoop
33 {
34     Q_OBJECT
35 public:
36     explicit EventLoopWithExitStatus(QObject * parent = nullptr);

```

```

37
38     enum class ExitStatus
39     {
40         Success,
41         Failure,
42         Timeout
43     };
44
45     friend QDebug & operator«(QDebug & dbg, const ExitStatus status);
46
47     friend QDataStream & operator«(
48         QDataStream & strm, const ExitStatus status);
49
50     ExitStatus exitStatus() const;
51     const QString & errorDescription() const;
52
53 public Q_SLOTS:
54     void exitAsSuccess();
55     void exitAsFailure();
56     void exitAsFailureWithError(QString errorDescription);
57     void exitAsFailureWithError(QString errorDescription);
58     void exitAsTimeout();
59
60 private:
61     ExitStatus m_exitStatus;
62     QString m_errorDescription;
63 };
64
65 } // namespace quentier
66
67 #endif // LIB_QUENTIER_UTILITY_EVENT_LOOP_WITH_EXIT_STATUS_H

```

## 6.59 FileCopier.h

```

1 /*
2  * Copyright 2018-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_FILE_COPIER_H
20 #define LIB_QUENTIER_UTILITY_FILE_COPIER_H
21
22 #include <quentier/types/ErrorMessage.h>
23 #include <quentier/utility/Linkage.h>
24
25 #include <QObject>
26 #include <QString>
27
28 QT_FORWARD_DECLARE_CLASS(QDebug)
29 QT_FORWARD_DECLARE_CLASS(QDataStream)
30
31 namespace quentier {
32
33 QT_FORWARD_DECLARE_CLASS(FileCopierPrivate)
34
35 class QUENTIER_EXPORT FileCopier : public QObject
36 {
37     Q_OBJECT
38 public:
39     explicit FileCopier(QObject * parent = nullptr);
40
41     enum class State
42     {
43         Idle = 0,
44         Copying,
45         Cancelling
46     };
47
48     friend QDebug & operator«(QDebug & dbg, const State state);
49     friend QDataStream & operator«(QDataStream & strm, const State state);

```

```

50
51     State state() const;
52
53     QString sourceFilePath() const;
54     QString destinationFilePath() const;
55
56     double currentProgress() const;
57
58 Q_SIGNALS:
59     void progressUpdate(double progress);
60     void finished(QString sourcePath, QString destPath);
61     void cancelled(QString sourcePath, QString destPath);
62     void notifyError(ErrorString error);
63
64 public Q_SLOTS:
65     void copyFile(QString sourcePath, QString destPath);
66     void cancel();
67
68 private:
69     Q_DISABLE_COPY(FileCopier)
70
71 private:
72     FileCopierPrivate * d_ptr;
73     Q_DECLARE_PRIVATE(FileCopier)
74 };
75
76 } // namespace quentier
77
78 #endif // LIB_QUENTIER_UTILITY_FILE_COPIER_H

```

## 6.60 FileIOProcessorAsync.h

```

1 /*
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_FILE_IO_PROCESSOR_ASYNC_H
20 #define LIB_QUENTIER_UTILITY_FILE_IO_PROCESSOR_ASYNC_H
21
22 #include <quentier/types/ErrorString.h>
23 #include <quentier/utility/Linkage.h>
24
25 #include <QByteArray>
26 #include <QIODevice>
27 #include <QObject>
28 #include <QString>
29 #include <QUuid>
30
31 namespace quentier {
32
33 QT_FORWARD_DECLARE_CLASS(FileIOProcessorAsyncPrivate)
34
35
36 class QUINTIER_EXPORT FileIOProcessorAsync : public QObject
37 {
38     Q_OBJECT
39 public:
40     explicit FileIOProcessorAsync(QObject * parent = nullptr);
41
42     void setIdleTimePeriod(qint32 seconds);
43
44 Q_SIGNALS:
45     void readyForIO();
46
47     void writeFileRequestProcessed(
48         bool success, ErrorString errorDescription, QUuid requestId);
49
50     void readFileRequestProcessed(
51         bool success, ErrorString errorDescription, QByteArray data,

```

```

91         QUuid requestId);
92
93 public Q_SLOTS:
94     void onWriteFileRequest(
95         QString absoluteFilePath, QByteArray data, QUuid requestId,
96         bool append);
97
98     void onReadFileRequest(QString absoluteFilePath, QUuid requestId);
99
100 private:
101     FileIOProcessorAsyncPrivate * const d_ptr;
102     Q_DECLARE_PRIVATE(FileIOProcessorAsync)
103 };
104
105 } // namespace quentier
106
107 #endif // LIB_QUENTIER_UTILITY_FILE_IO_PROCESSOR_ASYNC_H

```

## 6.61 FileSystem.h

```

1 /*
2  * Copyright 2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_FILE_SYSTEM_H
20 #define LIB_QUENTIER_UTILITY_FILE_SYSTEM_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QString>
25
26 namespace quentier {
27
28     QT_FORWARD_DECLARE_CLASS(ErrorString)
29
30
31     const QString QUENTIER_EXPORT relativePathFromAbsolutePath(
32         const QString & absolutePath, const QString & relativePathRootFolderPath);
33
34     bool QUENTIER_EXPORT removeFile(const QString & filePath);
35
36     bool QUENTIER_EXPORT removeDir(const QString & dirPath);
37
38     QByteArray QUENTIER_EXPORT
39     readFileContents(const QString & filePath, ErrorString & errorDescription);
40
41     bool QUENTIER_EXPORT renameFile(
42         const QString & from, const QString & to, ErrorString & errorDescription);
43
44 } // namespace quentier
45
46 #endif // LIB_QUENTIER_UTILITY_FILE_SYSTEM_H

```

## 6.62 FileSystemWatcher.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,

```

```

11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_FILE_SYSTEM_WATCHER_H
20 #define LIB_QUENTIER_UTILITY_FILE_SYSTEM_WATCHER_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QObject>
25 #include <QStringList>
26
27 #define FILE_SYSTEM_WATCHER_DEFAULT_REMOVAL_TIMEOUT_MSEC (500)
28
29 namespace quentier {
30
31 QT_FORWARD_DECLARE_CLASS(FileSystemWatcherPrivate)
32
33 class QUENTIER_EXPORT FileSystemWatcher : public QObject
34 {
35     Q_OBJECT
36 public:
37     explicit FileSystemWatcher(
38         const int removalTimeoutMsec =
39             FILE_SYSTEM_WATCHER_DEFAULT_REMOVAL_TIMEOUT_MSEC,
40         QObject * parent = nullptr);
41
42     explicit FileSystemWatcher(
43         const QStringList & paths,
44         const int removalTimeoutMsec =
45             FILE_SYSTEM_WATCHER_DEFAULT_REMOVAL_TIMEOUT_MSEC,
46         QObject * parent = nullptr);
47
48     virtual ~FileSystemWatcher() override;
49
50     void addPath(const QString & path);
51     void addPaths(const QStringList & paths);
52
53     QStringList directories() const;
54     QStringList files() const;
55
56     void removePath(const QString & path);
57     void removePaths(const QStringList & paths);
58
59 Q_SIGNALS:
60     void directoryChanged(const QString & path);
61     void directoryRemoved(const QString & path);
62
63     void fileChanged(const QString & path);
64     void fileRemoved(const QString & path);
65
66 private:
67     Q_DISABLE_COPY(FileSystemWatcher)
68
69 private:
70     FileSystemWatcherPrivate * d_ptr;
71     Q_DECLARE_PRIVATE(FileSystemWatcher)
72 };
73
74 } // namespace quentier
75
76 #endif // LIB_QUENTIER_UTILITY_FILE_SYSTEM_WATCHER_H

```

## 6.63 IKeychainService.h

```

1 /*
2 * Copyright 2018-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *

```

```

15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_I_KEYCHAIN_SERVICE_H
20 #define LIB_QUENTIER_UTILITY_I_KEYCHAIN_SERVICE_H
21
22 #include <quentier/types/ErrorMessage.h>
23 #include <quentier/utility/ForwardDeclarations.h>
24 #include <quentier/utility/Linkage.h>
25
26 #include <QObject>
27 #include <QUuid>
28
29 #include <memory>
30
31 QT_FORWARD_DECLARE_CLASS(QDebug)
32
33 namespace quentier {
34
35 class QUENTIER_EXPORT IKeychainService : public QObject
36 {
37     Q_OBJECT
38 protected:
39     explicit IKeychainService(QObject * parent = nullptr);
40
41 public:
42     virtual ~IKeychainService() {}
43
44     enum class ErrorCode
45     {
46         NoError,
47         EntryNotFound,
48         CouldNotDeleteEntry,
49         AccessDeniedByUser,
50         AccessDenied,
51         NoBackendAvailable,
52         NotImplemented,
53         OtherError
54     };
55
56     friend QTextStream & operator<<(
57         QTextStream & strm, const ErrorCode errorCode);
58
59     friend QDebug & operator<<(QDebug & dbg, const ErrorCode errorCode);
60
61 public:
62     virtual QUuid startWritePasswordJob(
63         const QString & service, const QString & key,
64         const QString & password) = 0;
65
66     virtual QUuid startReadPasswordJob(
67         const QString & service, const QString & key) = 0;
68
69     virtual QUuid startDeletePasswordJob(
70         const QString & service, const QString & key) = 0;
71
72 Q_SIGNALS:
73     void writePasswordJobFinished(
74         QUuid requestId, ErrorCode errorCode, ErrorMessage errorDescription);
75
76     void readPasswordJobFinished(
77         QUuid requestId, ErrorCode errorCode, ErrorMessage errorDescription,
78         QString password);
79
80     void deletePasswordJobFinished(
81         QUuid requestId, ErrorCode errorCode, ErrorMessage errorDescription);
82
83 private:
84     Q_DISABLE_COPY(IKeychainService);
85 };
86
87 QUENTIER_EXPORT IKeychainServicePtr
88 newQtKeychainService(QObject * parent = nullptr);
89
90 QUENTIER_EXPORT IKeychainServicePtr
91 newObfuscatingKeychainService(QObject * parent = nullptr);
92
93 QUENTIER_EXPORT IKeychainServicePtr newCompositeKeychainService(
94     QString name, IKeychainServicePtr primaryKeychain,
95     IKeychainServicePtr secondaryKeychain, QObject * parent = nullptr);
96
97 QUENTIER_EXPORT IKeychainServicePtr newMigratingKeychainService(
98     IKeychainServicePtr sourceKeychain, IKeychainServicePtr sinkKeychain,
99     QObject * parent = nullptr);
100
101 } // namespace quentier

```

```

205
206 #endif // LIB_QUENTIER_UTILITY_I_KEYCHAIN_SERVICE_H

```

## 6.64 Initialize.h

```

1 /*
2 * Copyright 2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_INITIALIZE_H
20 #define LIB_QUENTIER_UTILITY_INITIALIZE_H
21
22 #include <quentier/utility/Linkage.h>
23
24 namespace quentier {
25
26 void QUINTIER_EXPORT initializeLibquentier();
27
28 } // namespace quentier
29
30 #endif // LIB_QUENTIER_UTILITY_INITIALIZE_H

```

## 6.65 LimitedStack.h

```

1 /*
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_LIMITED_STACK_H
20 #define LIB_QUENTIER_UTILITY_LIMITED_STACK_H
21
22 #include <QStack>
23
24 namespace quentier {
25
26 template <class T>
27 class LimitedStack : public QStack<T>
28 {
29 public:
30     LimitedStack(void (*deleter)(T &) = nullptr) :
31         m_limit(-1), m_deleter(deleter)
32     {}
33
34     LimitedStack(const LimitedStack<T> & other) :
35         QStack<T>(other), m_limit(other.m_limit), m_deleter(other.m_deleter)
36     {}
37
38     LimitedStack(LimitedStack<T> && other) :
39         QStack<T>(std::move(other)), m_limit(std::move(other.m_limit)),
40         m_deleter(std::move(other.m_deleter))
41     {}
42
43 }
44
45 #endif

```

```

48
49     LimitedStack<T> & operator=(const LimitedStack<T> & other)
50     {
51         if (this != &other) {
52             QStack<T>::operator=(other);
53             m_limit = other.m_limit;
54             m_deleter = other.m_deleter;
55         }
56
57         return *this;
58     }
59
60     LimitedStack<T> & operator=(LimitedStack<T> && other)
61     {
62         if (this != &other) {
63             QStack<T>::operator=(std::move(other));
64             m_limit = std::move(other.m_limit);
65             m_deleter = std::move(other.m_deleter);
66         }
67
68         return *this;
69     }
70
71     ~LimitedStack()
72     {
73         if (m_deleter) {
74             while (!QStack<T>::isEmpty()) {
75                 T t = QStack<T>::pop();
76                 (*m_deleter)(t);
77             }
78         }
79     }
80
81     void swap(const LimitedStack<T> & other)
82     {
83         int limit = other.m_limit;
84         other.m_limit = m_limit;
85         m_limit = limit;
86
87         void (*deleter)(T &) = other.m_deleter;
88         other.m_deleter = m_deleter;
89         m_deleter = deleter;
90
91         QStack<T>::swap(other);
92     }
93
94     int limit()const
95     {
96         return m_limit;
97     }
98     void setLimit(const int limit)
99     {
100         m_limit = limit;
101     }
102
103     void push(const T & t)
104     {
105         if ((m_limit > 0) && (QVector<T>::size() == m_limit - 1)) {
106             if (m_deleter) {
107                 (*m_deleter)(*QVector<T>::begin());
108             }
109             Q_UNUSED(QVector<T>::erase(QVector<T>::begin()));
110         }
111         QStack<T>::push(t);
112     }
113
114 private:
115     int m_limit;
116     void (*m_deleter)(T &);
117 };
118
119 } // namespace quentier
120
121 #endif // LIB_QUENTIER_UTILITY_LIMITED_STACK_H

```

## 6.66 Linkage.h

```

1 /*
2 * Copyright 2016-2019 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *

```



```

6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_LINKAGE_H
20 #define LIB_QUENTIER_UTILITY_LINKAGE_H
21
22 #include <QtGlobal>
23
24 #if defined(BUILDING_QUENTIER_DLL)
25 #define QUENTIER_EXPORT Q_DECL_EXPORT
26 #else
27 #define QUENTIER_EXPORT Q_DECL_IMPORT
28 #endif
29
30 #endif // LIB_QUENTIER_UTILITY_LINKAGE_H

```

## 6.67 LRUCache.hpp

```

1 /*
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_LRU_CACHE_HPP
20 #define LIB_QUENTIER_UTILITY_LRU_CACHE_HPP
21
22 #include <QHash>
23
24 #include <cstdint>
25 #include <list>
26
27 namespace quentier {
28
29 template <
30     class Key, class Value,
31     class Allocator = std::allocator<std::pair<Key, Value>>
32 class LRUCache
33 {
34 public:
35     LRUCache(const size_t maxSize = 100) :
36         m_container(), m_currentSize(0), m_maxSize(maxSize), m_mapper()
37     {}
38
39     using key_type = Key;
40     using mapped_type = Value;
41     using allocator_type = Allocator;
42     using value_type = std::pair<key_type, mapped_type>;
43     using container_type = std::list<value_type, allocator_type>;
44     using size_type = typename container_type::size_type;
45     using difference_type = typename container_type::difference_type;
46     using iterator = typename container_type::iterator;
47     using const_iterator = typename container_type::const_iterator;
48     using reverse_iterator = std::reverse_iterator<iterator>;
49     using const_reverse_iterator = std::reverse_iterator<const_iterator>;
50
51     using reference = value_type &;
52     using const_reference = const value_type &;
53     using pointer = typename std::allocator_traits<allocator_type>::pointer;
54
55     using const_pointer =

```

```

56         typename std::allocator_traits<allocator_type>::const_pointer;
57
58     iterator begin()
59     {
60         return m_container.begin();
61     }
62     const_iterator begin() const
63     {
64         return m_container.begin();
65     }
66
67     reverse_iterator rbegin()
68     {
69         return m_container.rbegin();
70     }
71     const_reverse_iterator rbegin() const
72     {
73         return m_container.rbegin();
74     }
75
76     iterator end()
77     {
78         return m_container.end();
79     }
80     const_iterator end() const
81     {
82         return m_container.end();
83     }
84
85     reverse_iterator rend()
86     {
87         return m_container.rend();
88     }
89     const_reverse_iterator rend() const
90     {
91         return m_container.rend();
92     }
93
94     bool empty() const
95     {
96         return m_container.empty();
97     }
98     size_t size() const
99     {
100         return m_currentSize;
101     }
102     size_t max_size() const
103     {
104         return m_maxSize;
105     }
106
107     void clear()
108     {
109         m_container.clear();
110         m_mapper.clear();
111         m_currentSize = 0;
112     }
113
114     void put(const key_type & key, const mapped_type & value)
115     {
116         Q_UNUSED(remove(key))
117
118         m_container.push_front(value_type(key, value));
119         m_mapper[key] = m_container.begin();
120         ++m_currentSize;
121
122         fixupSize();
123     }
124
125     const mapped_type * get(const key_type & key) const
126     {
127         auto mapperIt = m_mapper.find(key);
128         if (mapperIt == m_mapper.end()) {
129             return nullptr;
130         }
131
132         auto it = mapperIt.value();
133         if (it == m_container.end()) {
134             return nullptr;
135         }
136
137         m_container.splice(m_container.begin(), m_container, it);
138         mapperIt.value() = m_container.begin();
139         return &(mapperIt.value()->second);
140     }
141
142     bool exists(const key_type & key)

```

```

143     {
144         auto mapperIt = m_mapper.find(key);
145         if (mapperIt == m_mapper.end()) {
146             return false;
147         }
148
149         auto it = mapperIt.value();
150         return (it != m_container.end());
151     }
152
153     bool remove(const key_type & key)
154     {
155         auto mapperIt = m_mapper.find(key);
156         if (mapperIt == m_mapper.end()) {
157             return false;
158         }
159
160         auto it = mapperIt.value();
161         Q_UNUSED(m_container.erase(it))
162         Q_UNUSED(m_mapper.erase(mapperIt))
163
164         if (m_currentSize != 0) {
165             --m_currentSize;
166         }
167
168         return true;
169     }
170
171     void setMaxSize(const size_t maxSize)
172     {
173         if (maxSize >= m_maxSize) {
174             m_maxSize = maxSize;
175             return;
176         }
177
178         size_t diff = m_maxSize - maxSize;
179         for (size_t i = 0; (i < diff) && !m_container.empty(); ++i) {
180             auto lastIt = m_container.end();
181             --lastIt;
182
183             const key_type & lastElementKey = lastIt->first;
184             Q_UNUSED(m_mapper.remove(lastElementKey))
185             Q_UNUSED(m_container.erase(lastIt))
186
187             if (m_currentSize != 0) {
188                 --m_currentSize;
189             }
190         }
191     }
192
193 private:
194     void fixupSize()
195     {
196         if (m_currentSize <= m_maxSize) {
197             return;
198         }
199
200         if (Q_UNLIKELY(m_container.empty())) {
201             return;
202         }
203
204         auto lastIt = m_container.end();
205         --lastIt;
206
207         const key_type & lastElementKey = lastIt->first;
208
209         Q_UNUSED(m_mapper.remove(lastElementKey))
210         Q_UNUSED(m_container.erase(lastIt))
211
212         if (m_currentSize != 0) {
213             --m_currentSize;
214         }
215     }
216
217 private:
218     mutable container_type m_container;
219     size_t m_currentSize;
220     size_t m_maxSize;
221
222     mutable QHash<Key, iterator> m_mapper;
223 };
224
225 } // namespace quentier
226
227 #endif // LIB_QUENTIER_UTILITY_LRU_CACHE_HPP

```

## 6.68 MessageBox.h

```

1 /*
2  * Copyright 2017-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_MESSAGE_BOX_H
20 #define LIB_QUENTIER_UTILITY_MESSAGE_BOX_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QMessageBox>
25
26 namespace quentier {
27
28 int QUINTIER_EXPORT genericMessageBox(
29     QWidget * parent, const QString & title, const QString & briefText,
30     const QString & detailedText = {},
31     const QMessageBox::StandardButtons standardButtons = QMessageBox::Ok);
32
33 int QUINTIER_EXPORT informationMessageBox(
34     QWidget * parent, const QString & title, const QString & briefText,
35     const QString & detailedText = {},
36     const QMessageBox::StandardButtons standardButtons = QMessageBox::Ok);
37
38 int QUINTIER_EXPORT warningMessageBox(
39     QWidget * parent, const QString & title, const QString & briefText,
40     const QString & detailedText = {},
41     const QMessageBox::StandardButtons standardButtons = QMessageBox::Ok);
42
43 int QUINTIER_EXPORT criticalMessageBox(
44     QWidget * parent, const QString & title, const QString & briefText,
45     const QString & detailedText = {},
46     const QMessageBox::StandardButtons standardButtons = QMessageBox::Ok);
47
48 int QUINTIER_EXPORT questionMessageBox(
49     QWidget * parent, const QString & title, const QString & briefText,
50     const QString & detailedText = {},
51     const QMessageBox::StandardButtons standardButtons = QMessageBox::Ok |
52         QMessageBox::Cancel);
53
54 void QUINTIER_EXPORT
55     internalErrorMessageBox(QWidget * parent, QString detailedText = {});
56 } // namespace quentier
57
58 #endif // LIB_QUENTIER_UTILITY_MESSAGE_BOX_H

```

## 6.69 Printable.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_PRINTABLE_H

```

```

20 #define LIB_QUENTIER_UTILITY_PRINTABLE_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QDebug>
25 #include <QHash>
26 #include <QSet>
27 #include <QString>
28 #include <QTextStream>
29
30 namespace quentier {
31
32 class QUENTIER_EXPORT Printable
33 {
34 public:
35     virtual QTextStream & print(QTextStream & strm) const = 0;
36
37     virtual const QString toString() const;
38
39     friend QUENTIER_EXPORT QTextStream & operator<<
40         (QTextStream & strm, const Printable & printable);
41
42     friend QUENTIER_EXPORT QDebug & operator<<
43         (QDebug & debug, const Printable & printable);
44
45 protected:
46     Printable();
47     Printable(const Printable & other);
48     Printable & operator=(const Printable & other);
49     virtual ~Printable();
50 };
51
52 } // namespace quentier
53
54 // printing operators for existing classes not inheriting from Printable
55
56 template <class T>
57 const QString ToString(const T & object)
58 {
59     QString str;
60     QTextStream strm(&str, QIODevice::WriteOnly);
61     strm << object;
62     return str;
63 }
64
65 template <class TKey, class TValue>
66 const QString ToString(const QHash<TKey, TValue> & object)
67 {
68     QString str;
69     QTextStream strm(&str, QIODevice::WriteOnly);
70     strm << QStringLiteral("QHash:  \n");
71
72     typedef typename QHash<TKey, TValue>::const_iterator CIter;
73     CIter hashEnd = object.end();
74     for (CIter it = object.begin(); it != hashEnd; ++it) {
75         strm << QStringLiteral("[") << it.key() << QStringLiteral("] = ")
76             << it.value() << QStringLiteral("; \n");
77     }
78     return str;
79 }
80
81 template <class T>
82 const QString ToString(const QSet<T> & object)
83 {
84     QString str;
85     QTextStream strm(&str, QIODevice::WriteOnly);
86     strm << QStringLiteral("QSet:  \n");
87
88     typedef typename QSet<T>::const_iterator CIter;
89     CIter setEnd = object.end();
90     for (CIter it = object.begin(); it != setEnd; ++it) {
91         strm << QStringLiteral("[") << *it << QStringLiteral("; \n");
92     }
93     return str;
94 }
95
96 #define QUENTIER_DECLARE_PRINTABLE(type, ...)
97 \
98 QUENTIER_EXPORT QTextStream & operator<<
99 (QTextStream & strm, const type & obj);
100 \
101 inline QDebug & operator<<(QDebug & debug, const type & obj)
102 {
103     debug << ToString<type, ##__VA_ARGS__>(obj);
104     return debug;
105 }
106 \
107 // QUENTIER_DECLARE_PRINTABLE
108
109
110

```

```
111 #endif // LIB_QUENTIER_UTILITY_PRINTABLE_H
```

## 6.70 QuentierApplication.h

```
1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_QUENTIER_APPLICATION_H
20 #define LIB_QUENTIER_UTILITY_QUENTIER_APPLICATION_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QApplication>
25
26 namespace quentier {
27
28 class QUINTIER_EXPORT QuentierApplication : public QApplication
29 {
30     Q_OBJECT
31 public:
32     QuentierApplication(int & argc, char * argv[]);
33     virtual ~QuentierApplication() override;
34
35     virtual bool notify(QObject * pObject, QEvent * pEvent) override;
36     virtual bool event(QEvent * pEvent) override;
37 };
38
39 } // namespace quentier
40
41 #endif // LIB_QUENTIER_UTILITY_QUENTIER_APPLICATION_H
```

## 6.71 QuentierCheckPtr.h

```
1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_QUENTIER_CHECK_PTR_H
20 #define LIB_QUENTIER_UTILITY_QUENTIER_CHECK_PTR_H
21
22 #include <quentier/exception/NullPtrException.h>
23 #include <quentier/logging/QuentierLogger.h>
24
25 #ifndef QUINTIER_CHECK_PTR
26 #define QUINTIER_CHECK_PTR(component, pointer, ...)
27 {
28     if (Q_UNLIKELY(!pointer)) {
29         using quentier::NullPtrException;
30         ErrorString quentier_null_ptr_error(
31             component,
32             pointer,
33             ...)
```

```

31 QT_TRANSLATE_NOOP("", "Detected unintended null pointer")); \
32     quentier_null_ptr_error.details() = QStringLiteral(__FILE__); \
33     quentier_null_ptr_error.details() += QStringLiteral(" "); \
34     quentier_null_ptr_error.details() += QString::number(__LINE__); \
35     quentier_null_ptr_error.details() += QStringLiteral(" "); \
36     quentier_null_ptr_error.details() += \
37         QString::fromUtf8("__VA_ARGS__"); \
38     QNERROR(component, quentier_null_ptr_error); \
39     throw NullPtrException(quentier_null_ptr_error); \
40 } \
41 } \
42 // QUENTIER_CHECK_PTR
43 #endif
44
45 #endif // LIB_QUENTIER_UTILITY_QUENTIER_CHECK_PTR_H

```

## 6.72 QuentierUndoCommand.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_QUENTIER_UNDO_COMMAND_H
20 #define LIB_QUENTIER_UTILITY_QUENTIER_UNDO_COMMAND_H
21
22 #include <quentier/types/ErrorMessage.h>
23
24 #include <QObject>
25 #include <QUndoCommand>
26
27 namespace quentier {
28
29     class QuentierUndoCommand : public QObject, public QUndoCommand
30     {
31     public:
32         QuentierUndoCommand(QUndoCommand * parent = nullptr);
33         QuentierUndoCommand(const QString & text, QUndoCommand * parent = nullptr);
34         virtual ~QuentierUndoCommand() override;
35
36         virtual void undo() override final;
37         virtual void redo() override final;
38
39         bool onceUndoExecuted() const
40         {
41             return m_onceUndoExecuted;
42         }
43
44     private:
45         void notifyError(ErrorMessage error);
46
47     protected:
48         virtual void undoImpl() = 0;
49         virtual void redoImpl() = 0;
50
51     private:
52         bool m_onceUndoExecuted;
53     };
54 } // namespace quentier
55
56 #endif // LIB_QUENTIER_UTILITY_QUENTIER_UNDO_COMMAND_H

```

## 6.73 ShortcutManager.h

```

1 /*

```

```
2 * Copyright 2016-2020 Dmitry Ivanov
3 *
4 * This file is part of libquentier
5 *
6 * libquentier is free software; you can redistribute it and/or modify
7 * it under the terms of the GNU Lesser General Public License as published by
8 * the Free Software Foundation, version 3 of the License.
9 *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_SHORTCUT_MANAGER_H
20 #define LIB_QUENTIER_UTILITY_SHORTCUT_MANAGER_H
21
22 #include <quentier/types/Account.h>
23 #include <quentier/utility/Linkage.h>
24
25 #include <QKeySequence>
26 #include <QObject>
27
28 namespace quentier {
29
30 QT_FORWARD_DECLARE_CLASS(ShortcutManagerPrivate)
31
32 class QUENTIER_EXPORT ShortcutManager : public QObject
33 {
34     Q_OBJECT
35 public:
36     explicit ShortcutManager(QObject * parent = nullptr);
37
38     enum QuentierShortcutKey
39     {
40         NewNote = 5000,
41         NewTag,
42         NewNotebook,
43         NewSavedSearch,
44         AddAttachment,
45         SaveAttachment,
46         OpenAttachment,
47         CopyAttachment,
48         CutAttachment,
49         RemoveAttachment,
50         RenameAttachment,
51         AddAccount,
52         ExitAccount,
53         SwitchAccount,
54         AccountInfo,
55         NoteSearch,
56         NewNoteSearch,
57         ShowNotes,
58         ShowNotebooks,
59         ShowTags,
60         ShowSavedSearches,
61         ShowDeletedNotes,
62         ShowStatusBar,
63         ShowToolBar,
64         PasteUnformatted,
65         Font,
66         UpperIndex,
67         LowerIndex,
68         AlignLeft,
69         AlignCenter,
70         AlignRight,
71         AlignFull,
72         IncreaseIndentation,
73         DecreaseIndentation,
74         IncreaseFontSize,
75         DecreaseFontSize,
76         InsertNumberedList,
77         InsertBulletedList,
78         Strikethrough,
79         Highlight,
80         InsertTable,
81         InsertRow,
82         InsertColumn,
83         RemoveRow,
84         RemoveColumn,
85         InsertHorizontalLine,
86         InsertToDoTag,
87         EditHyperlink,
88         CopyHyperlink,
```



```

89     RemoveHyperlink,
90     Encrypt,
91     Decrypt,
92     DecryptPermanently,
93     BackupLocalStorage,
94     RestoreLocalStorage,
95     UpgradeLocalStorage,
96     LocalStorageStatus,
97     SpellCheck,
98     SpellCheckIgnoreWord,
99     SpellCheckAddWordToUserDictionary,
100    SaveImage,
101    AnnotateImage,
102    ImageRotateClockwise,
103    ImageRotateCounterClockwise,
104    Synchronize,
105    FullSync,
106    ImportFolders,
107    Preferences,
108    ReleaseNotes,
109    ViewLogs,
110    About,
111    UnknownKey = 100000
112 };
113
114 QKeySequence shortcut (
115     const int key, const Account & account,
116     const QString & context = {}) const;
117
118 QKeySequence shortcut (
119     const QString & nonStandardKey, const Account & account,
120     const QString & context = {}) const;
121
122 QKeySequence defaultShortcut (
123     const int key, const Account & account,
124     const QString & context = {}) const;
125
126 QKeySequence defaultShortcut (
127     const QString & nonStandardKey, const Account & account,
128     const QString & context = {}) const;
129
130 QKeySequence userShortcut (
131     const int key, const Account & account,
132     const QString & context = {}) const;
133
134 QKeySequence userShortcut (
135     const QString & nonStandardKey, const Account & account,
136     const QString & context = {}) const;
137
138 Q_SIGNALS:
139     void shortcutChanged(
140         int key, QKeySequence shortcut, const Account & account,
141         QString context);
142
143     void nonStandardShortcutChanged(
144         QString nonStandardKey, QKeySequence shortcut, const Account & account,
145         QString context);
146
147 public Q_SLOTS:
148     void setUserShortcut (
149         int key, QKeySequence shortcut, const Account & account,
150         QString context = {});
151
152     void setNonStandardUserShortcut (
153         QString nonStandardKey, QKeySequence shortcut, const Account & account,
154         QString context = {});
155
156     void setDefaultShortcut (
157         int key, QKeySequence shortcut, const Account & account,
158         QString context = {});
159
160     void setNonStandardDefaultShortcut (
161         QString nonStandardKey, QKeySequence shortcut, const Account & account,
162         QString context = {});
163
164 private:
165     ShortcutManagerPrivate * const d_ptr;
166     Q_DECLARE_PRIVATE(ShortcutManager)
167 };
168
169 } // namespace quantier
170
171 #endif // LIB_QUANTIER_UTILITY_SHORTCUT_MANAGER_H

```

## 6.74 Size.h

```

1 /*
2  * Copyright 2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_SIZE_H
20 #define LIB_QUENTIER_UTILITY_SIZE_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QString>
25
26 namespace quentier {
27
28     const QString QUENTIER_EXPORT humanReadableSize(const quint64 bytes);
29
30 } // namespace quentier
31
32 #endif // LIB_QUENTIER_UTILITY_SIZE_H

```

## 6.75 StandardPaths.h

```

1 /*
2  * Copyright 2017-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_STANDARD_PATHS_H
20 #define LIB_QUENTIER_UTILITY_STANDARD_PATHS_H
21
22 #include <quentier/types/Account.h>
23 #include <quentier/utility/Linkage.h>
24
25 #define LIBQUENTIER_PERSISTENCE_STORAGE_PATH \
26     "LIBQUENTIER_PERSISTENCE_STORAGE_PATH"
27
28 namespace quentier {
29
30     const QString QUENTIER_EXPORT
31     applicationPersistentStoragePath(bool * pNonStandardLocation = nullptr);
32
33     const QString QUENTIER_EXPORT
34     accountPersistentStoragePath(const Account & account);
35
36     const QString QUENTIER_EXPORT applicationTemporaryStoragePath();
37
38     const QString QUENTIER_EXPORT homePath();
39
40     const QString QUENTIER_EXPORT documentsPath();
41
42 } // namespace quentier
43
44 #endif // LIB_QUENTIER_UTILITY_STANDARD_PATHS_H

```

## 6.76 StringUtils.h

```

1  /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUESTIER_UTILITY_STRING_UTILS_H
20 #define LIB_QUESTIER_UTILITY_STRING_UTILS_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QSet>
25 #include <QString>
26 #include <QVector>
27
28 namespace quentier {
29
30 QT_FORWARD_DECLARE_CLASS(StringUtilsPrivate)
31
32 class QUESTIER_EXPORT StringUtils
33 {
34 public:
35     StringUtils();
36     virtual ~StringUtils();
37
38     void removePunctuation(
39         QString & str, const QVector<QChar> & charactersToPreserve = {}) const;
40
41     void removeDiacritics(QString & str) const;
42     void removeNewlines(QString & str) const;
43
44     struct StringFilterPredicate
45     {
46         StringFilterPredicate(QSet<QString> & filteredStrings) :
47             m_filteredStrings(filteredStrings)
48         {}
49
50         bool operator()(const QString & str) const
51     {
52         return m_filteredStrings.contains(str);
53     }
54
55     QSet<QString> & m_filteredStrings;
56     };
57
58 private:
59     StringUtilsPrivate * const d_ptr;
60     Q_DECLARE_PRIVATE(StringUtils);
61 };
62
63 } // namespace quentier
64
65 #endif // LIB_QUESTIER_UTILITY_STRING_UTILS_H

```

## 6.77 SuppressWarnings.h

```

1  /*
2  * Copyright 2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.

```

```

14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_SUPPRESS_WARNINGS_H
20 #define LIB_QUENTIER_UTILITY_SUPPRESS_WARNINGS_H
21
22 // Common macros
23
24 #define STRINGIFY(a) #a
25
26 // Define empty macros doing nothing for supported compilers, they would be used
27 // as fallback when any of these compilers are not actually used
28
29 #define SAVE_WARNINGS
30
31 #define CLANG_SUPPRESS_WARNING(warning)
32 #define GCC_SUPPRESS_WARNING(warning)
33 #define MSVC_SUPPRESS_WARNING(warning)
34
35 #define RESTORE_WARNINGS
36
37 // Clang implementation
38
39 #if defined(__clang__)
40
41 #undef CLANG_SUPPRESS_WARNING
42
43 #define CLANG_SUPPRESS_WARNING(warning) \
44     _Pragma( \
45         STRINGIFY(clang diagnostic ignored #warning)) // CLANG_IGNORE_WARNING
46
47 #undef SAVE_WARNINGS
48
49 #define SAVE_WARNINGS _Pragma("clang diagnostic push") // SAVE_WARNINGS
50
51 #undef RESTORE_WARNINGS
52
53 #define RESTORE_WARNINGS _Pragma("clang diagnostic pop") // RESTORE_WARNINGS
54
55 #endif // clang
56
57 // GCC implementation
58
59 // Clang can mimic gcc so need to ensure it's indeed gcc
60 #if defined(__GNUC__) && !defined(__clang__)
61
62 #undef GCC_SUPPRESS_WARNING
63
64 #define GCC_SUPPRESS_WARNING(warning) \
65     _Pragma(STRINGIFY(GCC diagnostic ignored #warning)) // GCC_SUPPRESS_WARNING
66
67 #undef SAVE_WARNINGS
68
69 #define SAVE_WARNINGS _Pragma("GCC diagnostic push") // SAVE_WARNINGS
70
71 #undef RESTORE_WARNINGS
72
73 #define RESTORE_WARNINGS _Pragma("GCC diagnostic pop") // RESTORE_WARNINGS
74
75 #endif // GCC
76
77 // MSVC implementation
78
79 #if defined(_MSC_VER)
80
81 #undef MSVC_SUPPRESS_WARNING
82
83 #define MSVC_SUPPRESS_WARNING(number) \
84     __pragma(warning(disable : number)) // MSVC_SUPPRESS_WARNING
85
86 #undef SAVE_WARNINGS
87
88 #define SAVE_WARNINGS __pragma(warning(push)) // SAVE_WARNINGS
89
90 #undef RESTORE_WARNINGS
91
92 #define RESTORE_WARNINGS __pragma(warning(pop)) // RESTORE_WARNINGS
93
94 #endif // MSVC
95
96 #endif // LIB_QUENTIER_UTILITY_SUPPRESS_WARNINGS_H

```

## 6.78 SysInfo.h

```

1  /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_SYS_INFO_H
20 #define LIB_QUENTIER_UTILITY_SYS_INFO_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QString>
25
26 namespace quentier {
27
28 QT_FORWARD_DECLARE_CLASS(SysInfoPrivate)
29
30 class QUINTIER_EXPORT SysInfo
31 {
32 public:
33     SysInfo();
34     ~SysInfo();
35
36     qint64 pageSize();
37     qint64 totalMemory();
38     qint64 freeMemory();
39
40     QString stackTrace();
41
42     QString platformName();
43
44 private:
45     Q_DISABLE_COPY(SysInfo)
46
47 private:
48     SysInfoPrivate * const d_ptr;
49     Q_DECLARE_PRIVATE(SysInfo)
50 };
51
52 } // namespace quentier
53
54 #endif // LIB_QUENTIER_UTILITY_SYS_INFO_H

```

## 6.79 System.h

```

1  /*
2  * Copyright 2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_SYSTEM_H
20 #define LIB_QUENTIER_UTILITY_SYSTEM_H
21
22 #include <quentier/utility/Linkage.h>
23
24 #include <QString>

```

```

25 #include <QUrl>
26
27 namespace quentier {
28
32 QString QUENTIER_EXPORT getCurrentUserName();
33
37 QString QUENTIER_EXPORT getCurrentUserFullName();
38
42 void QUENTIER_EXPORT openUrl(const QUrl & url);
43
44 } // namespace quentier
45
46 #endif // LIB_QUENTIER_UTILITY_SYSTEM_H

```

## 6.80 TagSortByParentChildRelations.h

```

1 /*
2  * Copyright 2017-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_TAG_SORT_BY_PARENT_CHILD_RELATIONS_H
20 #define LIB_QUENTIER_UTILITY_TAG_SORT_BY_PARENT_CHILD_RELATIONS_H
21
22 #include <quentier/types/ErrorMessage.h>
23 #include <quentier/types/Tag.h>
24
25 #include <qt5qevercloud/QEverCloud.h>
26
27 #include <QList>
28
29 namespace quentier {
30
42 bool QUENTIER_EXPORT sortTagsByParentChildRelations(
43     QList<qevercloud::Tag> & tagList, ErrorMessage & errorMessage);
44
57 bool QUENTIER_EXPORT sortTagsByParentChildRelations(
58     QList<Tag> & tagList, ErrorMessage errorMessage);
59
60 } // namespace quentier
61
62 #endif // LIB_QUENTIER_UTILITY_TAG_SORT_BY_PARENT_CHILD_RELATIONS_H

```

## 6.81 UidGenerator.h

```

1 /*
2  * Copyright 2016-2020 Dmitry Ivanov
3  *
4  * This file is part of libquentier
5  *
6  * libquentier is free software; you can redistribute it and/or modify
7  * it under the terms of the GNU Lesser General Public License as published by
8  * the Free Software Foundation, version 3 of the License.
9  *
10 * libquentier is distributed in the hope that it will be useful,
11 * but WITHOUT ANY WARRANTY; without even the implied warranty of
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 * GNU Lesser General Public License for more details.
14 *
15 * You should have received a copy of the GNU Lesser General Public License
16 * along with libquentier. If not, see <http://www.gnu.org/licenses/>.
17 */
18
19 #ifndef LIB_QUENTIER_UTILITY_UID_GENERATOR_H
20 #define LIB_QUENTIER_UTILITY_UID_GENERATOR_H
21

```

```
22 #include <quentier/utility/Linkage.h>
23
24 #include <QString>
25 #include <QUuid>
26
27 namespace quentier {
28
29 class QUINTIER_EXPORT UidGenerator
30 {
31 public:
32     static QString Generate();
33     static QString UidToString(const QUuid & uid);
34 };
35
36 } // namespace quentier
37
38 #endif // LIB_QUENTIER_UTILITY_UID_GENERATOR_H
```





# Index

- ~ApplicationSettings
  - quentier::ApplicationSettings, [17](#)
- AccessDenied
  - quentier::IKeychainService, [58](#)
- AccessDeniedByUser
  - quentier::IKeychainService, [58](#)
- Account.h, [275](#)
- accountHighUsn
  - quentier::LocalStorageManager, [109](#)
- active
  - quentier::SynchronizationManager, [211](#)
- addEnResource
  - quentier::LocalStorageManager, [110](#)
- addLinkedNotebook
  - quentier::LocalStorageManager, [110](#)
- addNote
  - quentier::LocalStorageManager, [111](#)
- addNotebook
  - quentier::LocalStorageManager, [111](#)
- addSavedSearch
  - quentier::LocalStorageManager, [112](#)
- addTag
  - quentier::LocalStorageManager, [112](#)
- addUser
  - quentier::LocalStorageManager, [112](#)
- ApplicationSettings
  - quentier::ApplicationSettings, [16](#), [17](#)
- ApplicationSettings.h, [301](#)
- ApplicationSettingsInitializationException.h, [230](#)
- apply
  - quentier::ILocalStoragePatch, [66](#)
- authenticate
  - quentier::SynchronizationManager, [211](#)
- authenticateCurrentAccount
  - quentier::SynchronizationManager, [212](#)
- authenticateToSharedNotebook
  - quentier::INoteStore, [74](#)
- authenticationFinished
  - quentier::SynchronizationManager, [212](#)
- AuthenticationManager.h, [267](#)
- authenticationRevoked
  - quentier::SynchronizationManager, [212](#)
- backend
  - quentier::NoteEditor, [167](#)
- backupLocalStorage
  - quentier::ILocalStoragePatch, [66](#)
- backupProgress
  - quentier::ILocalStoragePatch, [67](#)
- beginGroup
  - quentier::ApplicationSettings, [17](#), [18](#)
- beginReadArray
  - quentier::ApplicationSettings, [18](#)
- beginWriteArray
  - quentier::ApplicationSettings, [19](#)
- checkLinkedNotebooks
  - quentier::DefaultLocalStorageCacheExpiryChecker, [35](#)
  - quentier::ILocalStorageCacheExpiryChecker, [62](#)
- checkNotebooks
  - quentier::DefaultLocalStorageCacheExpiryChecker, [35](#)
  - quentier::ILocalStorageCacheExpiryChecker, [62](#)
- checkNotes
  - quentier::DefaultLocalStorageCacheExpiryChecker, [35](#)
  - quentier::ILocalStorageCacheExpiryChecker, [62](#)
- checkParameters
  - quentier::LinkedNotebook, [95](#)
  - quentier::Note, [156](#)
  - quentier::Notebook, [162](#)
  - quentier::Resource, [187](#)
  - quentier::SavedSearch, [195](#)
  - quentier::Tag, [222](#)
- checkResources
  - quentier::DefaultLocalStorageCacheExpiryChecker, [36](#)
  - quentier::ILocalStorageCacheExpiryChecker, [63](#)
- Checks.h, [302](#)
- checkSavedSearches
  - quentier::DefaultLocalStorageCacheExpiryChecker, [36](#)
  - quentier::ILocalStorageCacheExpiryChecker, [63](#)
- checkTags
  - quentier::DefaultLocalStorageCacheExpiryChecker, [36](#)
  - quentier::ILocalStorageCacheExpiryChecker, [63](#)
- checkVersion
  - quentier::IUserStore, [91](#)
- cleanupExternalHtml
  - quentier::ENMLConverter, [41](#)
- clear
  - quentier::LinkedNotebook, [96](#)
  - quentier::Note, [156](#)
  - quentier::Notebook, [162](#)
  - quentier::NoteEditor, [167](#)
  - quentier::Resource, [187](#)
  - quentier::SavedSearch, [195](#)

- quentier::Tag, [222](#)
- ClearDatabase
  - quentier::LocalStorageManager, [108](#)
- clone
  - quentier::DefaultLocalStorageCacheExpiryChecker, [36](#)
  - quentier::ILocalStorageCacheExpiryChecker, [63](#)
- Compat.h, [303](#)
- contains
  - quentier::ApplicationSettings, [20](#)
- convertToNote
  - quentier::NoteEditor, [168](#)
- CouldNotDeleteEntry
  - quentier::IKeychainService, [58](#)
- createNote
  - quentier::INoteStore, [74](#)
- createNotebook
  - quentier::INoteStore, [75](#)
- createSavedSearch
  - quentier::INoteStore, [75](#)
- createTag
  - quentier::INoteStore, [76](#)
- currentNoteLocalUid
  - quentier::NoteEditor, [168](#)
- DatabaseLockedException.h, [231](#)
- DatabaseLockFailedException.h, [231](#)
- DatabaseOpeningException.h, [232](#)
- DatabaseRequestException.h, [232](#)
- DateTime.h, [303](#)
- DecryptedTextManager.h, [227](#)
- defaultFont
  - quentier::NoteEditor, [168](#)
- DefaultLocalStorageCacheExpiryChecker.h, [237](#)
- defaultPalette
  - quentier::NoteEditor, [168](#)
- defaultShortcut
  - quentier::ShortcutManager, [203](#), [204](#)
- deletePasswordJobFinished
  - quentier::IKeychainService, [58](#)
- deleteUser
  - quentier::LocalStorageManager, [113](#)
- detectedConflictDuringLocalChangesSending
  - quentier::SynchronizationManager, [213](#)
- displayName
  - quentier::Account, [13](#)
- downloadNoteThumbnailsOption
  - quentier::SynchronizationManager, [213](#)
- EmptyDataElementException.h, [233](#)
- EncryptionManager.h, [304](#)
- ENMLConverter.h, [228](#)
- enResourceCount
  - quentier::LocalStorageManager, [113](#)
- EntryNotFound
  - quentier::IKeychainService, [58](#)
- ErrorCode
  - quentier::IKeychainService, [57](#)
- ErrorString.h, [277](#)
- EventLoopWithExitStatus.h, [305](#)
- evernoteAccountType
  - quentier::Account, [13](#)
- evernoteHost
  - quentier::Account, [13](#)
- exceptionDisplayName
  - quentier::ApplicationSettingsInitializationException, [24](#)
  - quentier::DatabaseLockedException, [28](#)
  - quentier::DatabaseLockFailedException, [30](#)
  - quentier::DatabaseOpeningException, [31](#)
  - quentier::DatabaseRequestException, [32](#)
  - quentier::EmptyDataElementException, [38](#)
  - quentier::LocalStorageCacheManagerException, [100](#)
  - quentier::LoggerInitializationException, [152](#)
  - quentier::NoteEditorInitializationException, [174](#)
  - quentier::NoteEditorPluginInitializationException, [176](#)
  - quentier::NullPtrException, [180](#)
- exportNotesToEnex
  - quentier::ENMLConverter, [42](#)
- expungeEnResource
  - quentier::LocalStorageManager, [114](#)
- expungeLinkedNotebook
  - quentier::LocalStorageManager, [114](#)
- expungeNote
  - quentier::LocalStorageManager, [115](#)
- expungeNotebook
  - quentier::LocalStorageManager, [115](#)
- expungeNotelessTagsFromLinkedNotebooks
  - quentier::LocalStorageManager, [116](#)
- expungeSavedSearch
  - quentier::LocalStorageManager, [116](#)
- expungeTag
  - quentier::LocalStorageManager, [116](#)
- expungeUser
  - quentier::LocalStorageManager, [117](#)
- failed
  - quentier::SynchronizationManager, [213](#)
- FileCopier.h, [306](#)
- FileIOProcessorAsync.h, [307](#)
- FileSystem.h, [308](#)
- FileSystemWatcher.h, [308](#)
- findDefaultNotebook
  - quentier::LocalStorageManager, [117](#)
- findDefaultOrLastUsedNotebook
  - quentier::LocalStorageManager, [118](#)
- findEnResource
  - quentier::LocalStorageManager, [118](#)
- findLastUsedNotebook
  - quentier::LocalStorageManager, [119](#)
- findLinkedNotebook
  - quentier::LocalStorageManager, [119](#)
- findNote
  - quentier::LocalStorageManager, [119](#)
- findNotebook
  - quentier::LocalStorageManager, [120](#)

- findNoteLocalUidsWithSearchQuery
  - quentier::LocalStorageManager, 120
- findNotesWithSearchQuery
  - quentier::LocalStorageManager, 121
- findSavedSearch
  - quentier::LocalStorageManager, 121
- findTag
  - quentier::LocalStorageManager, 122
- findUser
  - quentier::LocalStorageManager, 122
- finished
  - quentier::SynchronizationManager, 213
- ForwardDeclarations.h, 268
- fromVersion
  - quentier::ILocalStoragePatch, 67
- getAccountLimits
  - quentier::UserStore, 91
- getLinkedNotebookSyncChunk
  - quentier::INoteStore, 76
- getLinkedNotebookSyncState
  - quentier::INoteStore, 77
- getNote
  - quentier::INoteStore, 78
- getNoteAsync
  - quentier::INoteStore, 78
- GetNoteOption
  - quentier::LocalStorageManager, 107
- getResource
  - quentier::INoteStore, 79
- getResourceAsync
  - quentier::INoteStore, 80
- GetResourceOption
  - quentier::LocalStorageManager, 107
- getSyncChunk
  - quentier::INoteStore, 81
- getSyncState
  - quentier::INoteStore, 81
- getUser
  - quentier::IUserStore, 92
- guid
  - quentier::LinkedNotebook, 96
  - quentier::Note, 156
  - quentier::Notebook, 162
  - quentier::Resource, 187
  - quentier::SavedSearch, 195
  - quentier::Tag, 222
- hasGuid
  - quentier::LinkedNotebook, 96
  - quentier::Note, 156
  - quentier::Notebook, 162
  - quentier::Resource, 187
  - quentier::SavedSearch, 196
  - quentier::Tag, 222
- hasUpdateSequenceNumber
  - quentier::LinkedNotebook, 96
  - quentier::Note, 157
  - quentier::Notebook, 162
  - quentier::Resource, 188
  - quentier::SavedSearch, 196
  - quentier::Tag, 222
- highestSupportedLocalStorageVersion
  - quentier::LocalStorageManager, 123
- HTMLCleaner.h, 230
- htmlToQTextDocument
  - quentier::ENMLConverter, 42
- IAuthenticationManager.h, 269
- id
  - quentier::Account, 13
- idleTime
  - quentier::NoteEditor, 168
- IFavoritableDataElement.h, 277
- IKeychainService.h, 309
- ILocalStorageCacheExpiryChecker.h, 237
- ILocalStorageDataElement.h, 278
- ILocalStoragePatch.h, 238
- importEnex
  - quentier::ENMLConverter, 43
- inAppNoteLinkPasteRequested
  - quentier::NoteEditor, 168
- IncludeMilliseconds
  - quentier::DateTimePrint, 33
- IncludeNumericTimestamp
  - quentier::DateTimePrint, 33
- IncludeTimezone
  - quentier::DateTimePrint, 33
- initialize
  - quentier::NoteEditor, 169
- Initialize.h, 311
- INoteEditorBackend.h, 260
- INoteStore.h, 270
- INoteStoreDataElement.h, 279
- IQuentierException.h, 233
- isEditorPageModified
  - quentier::NoteEditor, 169
- isEmpty
  - quentier::Account, 13
- isLocalStorageVersionTooHigh
  - quentier::LocalStorageManager, 123
- isModified
  - quentier::NoteEditor, 169
- isNoteLoaded
  - quentier::NoteEditor, 170
- ISyncStateStorage.h, 272
- IUserStore.h, 272
- LimitedStack.h, 311
- Linkage.h, 312
- LinkedNotebook.h, 281
- linkedNotebookCount
  - quentier::LocalStorageManager, 123
- linkedNotebooksNotesDownloadProgress
  - quentier::SynchronizationManager, 214
- linkedNotebooksResourcesDownloadProgress
  - quentier::SynchronizationManager, 214
- linkedNotebooksSyncChunksDownloaded

- quentier::SynchronizationManager, 214
- linkedNotebookSyncChunksDownloadProgress
  - quentier::SynchronizationManager, 214
- listAllLinkedNotebooks
  - quentier::LocalStorageManager, 124
- listAllNotebooks
  - quentier::LocalStorageManager, 124
- listAllSavedSearches
  - quentier::LocalStorageManager, 125
- listAllSharedNotebooks
  - quentier::LocalStorageManager, 126
- listAllTags
  - quentier::LocalStorageManager, 126
- listAllTagsPerNote
  - quentier::LocalStorageManager, 127
- listLinkedNotebooks
  - quentier::LocalStorageManager, 127
- listNotebooks
  - quentier::LocalStorageManager, 128
- listNotes
  - quentier::LocalStorageManager, 129
- listNotesByLocalUids
  - quentier::LocalStorageManager, 129
- listNotesPerNotebook
  - quentier::LocalStorageManager, 130
- listNotesPerNotebooksAndTags
  - quentier::LocalStorageManager, 131
- listNotesPerTag
  - quentier::LocalStorageManager, 131
- ListObjectsOption
  - quentier::LocalStorageManager, 108
- Lists.h, 239
- listSavedSearches
  - quentier::LocalStorageManager, 132
- listSharedNotebooksPerNotebookGuid
  - quentier::LocalStorageManager, 133
- listTags
  - quentier::LocalStorageManager, 133
- listTagsWithNoteLocalUids
  - quentier::LocalStorageManager, 134
- LocalStorageCacheManager.h, 240
- LocalStorageCacheManagerException.h, 234
- LocalStorageManager
  - quentier::LocalStorageManager, 109
- LocalStorageManager.h, 241
- LocalStorageManagerAsync.h, 248
- localStorageRequiresUpgrade
  - quentier::LocalStorageManager, 135
- localStorageVersion
  - quentier::LocalStorageManager, 135
- LoggerInitializationException.h, 235
- LRUCache.hpp, 313
- MessageBox.h, 316
- name
  - quentier::Account, 14
- NoBackendAvailable
  - quentier::IKeychainService, 58
- NoError
  - quentier::IKeychainService, 58
- Note.h, 282
- Notebook.h, 285
- notebookCount
  - quentier::LocalStorageManager, 136
- notebookModifier
  - quentier::NoteSearchQuery, 178
- noteCount
  - quentier::LocalStorageManager, 136
- noteCountPerNotebook
  - quentier::LocalStorageManager, 136
- noteCountPerNotebooksAndTags
  - quentier::LocalStorageManager, 137
- noteCountPerTag
  - quentier::LocalStorageManager, 137
- noteCountsPerAllTags
  - quentier::LocalStorageManager, 138
- NoteEditor.h, 263
- NoteEditorInitializationException.h, 235
- NoteEditorPluginInitializationException.h, 236
- notesDownloadProgress
  - quentier::SynchronizationManager, 215
- NoteSearchQuery.h, 257
- noteStoreUrl
  - quentier::INoteStore, 82
- notifySyncStateUpdated
  - quentier::ISyncStateStorage, 90
- NotImplemented
  - quentier::IKeychainService, 58
- NullPtrException.h, 236
- onReadFileRequest
  - quentier::FileIOProcessorAsync, 49
- onWriteFileRequest
  - quentier::FileIOProcessorAsync, 50
- Option
  - quentier::DateTimePrint, 33
- OtherError
  - quentier::IKeychainService, 58
- OverrideLock
  - quentier::LocalStorageManager, 108
- patchLongDescription
  - quentier::ILocalStoragePatch, 67
- patchShortDescription
  - quentier::ILocalStoragePatch, 67
- preparedDirtyObjectsForSending
  - quentier::SynchronizationManager, 215
- preparedLinkedNotebooksDirtyObjectsForSending
  - quentier::SynchronizationManager, 215
- print
  - quentier::Account, 14
  - quentier::ApplicationSettings, 20
  - quentier::DefaultLocalStorageCacheExpiryChecker, 37
  - quentier::ENMLConverter::SkipHtmlElementRule, 207
  - quentier::ErrorString, 45

- quentier::ILocalStorageCacheExpiryChecker, 64
  - quentier::IQuentierException, 87
  - quentier::ISyncStateStorage::ISyncState, 88
  - quentier::LinkedNotebook, 96
  - quentier::LocalStorageCacheManager, 99
  - quentier::Note, 157
  - quentier::Notebook, 162
  - quentier::NoteSearchQuery, 179
  - quentier::Printable, 182
  - quentier::Resource, 188
  - quentier::ResourceRecognitionIndexItem, 191
  - quentier::ResourceRecognitionIndices, 192
  - quentier::SavedSearch, 196
  - quentier::SharedNote, 199
  - quentier::SharedNotebook, 202
  - quentier::Tag, 223
  - quentier::User, 226
- Printable.h, 316
- progress
  - quentier::ILocalStoragePatch, 67
- quentier::Account, 11
  - displayName, 13
  - evernoteAccountType, 13
  - evernoteHost, 13
  - id, 13
  - isEmpty, 13
  - name, 14
  - print, 14
  - setDisplayName, 14
  - shardId, 14
  - type, 14
- quentier::ApplicationSettings, 15
  - ~ApplicationSettings, 17
  - ApplicationSettings, 16, 17
  - beginGroup, 17, 18
  - beginReadArray, 18
  - beginWriteArray, 19
  - contains, 20
  - print, 20
  - remove, 20, 21
  - setValue, 21
  - value, 22
- quentier::ApplicationSettings::ArrayCloser, 25
- quentier::ApplicationSettings::GroupCloser, 52
- quentier::ApplicationSettingsInitializationException, 23
  - exceptionDisplayName, 24
- quentier::AuthenticationManager, 25
- quentier::DatabaseLockedException, 27
  - exceptionDisplayName, 28
- quentier::DatabaseLockFailedException, 29
  - exceptionDisplayName, 30
- quentier::DatabaseOpeningException, 30
  - exceptionDisplayName, 31
- quentier::DatabaseRequestException, 31
  - exceptionDisplayName, 32
- quentier::DateTimePrint, 33
  - IncludeMilliseconds, 33
  - IncludeNumericTimestamp, 33
  - IncludeTimezone, 33
  - Option, 33
- quentier::DecryptedTextManager, 33
- quentier::DefaultLocalStorageCacheExpiryChecker, 34
  - checkLinkedNotebooks, 35
  - checkNotebooks, 35
  - checkNotes, 35
  - checkResources, 36
  - checkSavedSearches, 36
  - checkTags, 36
  - clone, 36
  - print, 37
- quentier::EmptyDataElementException, 37
  - exceptionDisplayName, 38
- quentier::EncryptionManager, 39
- quentier::ENMLConverter, 40
  - cleanupExternalHtml, 41
  - exportNotesToEnex, 42
  - htmlToQTextDocument, 42
  - importEnex, 43
- quentier::ENMLConverter::NoteContentToHtmlExtraData, 163
- quentier::ENMLConverter::SkipHtmlElementRule, 205
  - print, 207
- quentier::ErrorString, 43
  - print, 45
- quentier::EventLoopWithExitStatus, 45
- quentier::FileCopier, 47
- quentier::FileIOProcessorAsync, 48
  - onReadFileRequest, 49
  - onWriteFileRequest, 50
  - readFileRequestProcessed, 50
  - setIdleTimePeriod, 50
  - writeFileRequestProcessed, 51
- quentier::FileSystemWatcher, 51
- quentier::HTMLCleaner, 53
- quentier::IAuthenticationManager, 54
- quentier::IFavoritableDataElement, 55
- quentier::IKeychainService, 56
  - AccessDenied, 58
  - AccessDeniedByUser, 58
  - CouldNotDeleteEntry, 58
  - deletePasswordJobFinished, 58
  - EntryNotFound, 58
  - ErrorCode, 57
  - NoBackendAvailable, 58
  - NoError, 58
  - NotImplemented, 58
  - OtherError, 58
  - readPasswordJobFinished, 58
  - startDeletePasswordJob, 59
  - startReadPasswordJob, 59
  - startWritePasswordJob, 59
  - writePasswordJobFinished, 60
- quentier::ILocalStorageCacheExpiryChecker, 60
  - checkLinkedNotebooks, 62
  - checkNotebooks, 62
  - checkNotes, 62

- checkResources, 63
- checkSavedSearches, 63
- checkTags, 63
- clone, 63
- print, 64
- quentier::ILocalStorageDataElement, 64
- quentier::ILocalStoragePatch, 65
  - apply, 66
  - backupLocalStorage, 66
  - backupProgress, 67
  - fromVersion, 67
  - patchLongDescription, 67
  - patchShortDescription, 67
  - progress, 67
  - removeLocalStorageBackup, 68
  - restoreBackupProgress, 68
  - restoreLocalStorageFromBackup, 68
  - toVersion, 69
- quentier::INoteEditorBackend, 69
- quentier::INoteStore, 72
  - authenticateToSharedNotebook, 74
  - createNote, 74
  - createNotebook, 75
  - createSavedSearch, 75
  - createTag, 76
  - getLinkedNotebookSyncChunk, 76
  - getLinkedNotebookSyncState, 77
  - getNote, 78
  - getNoteAsync, 78
  - getResource, 79
  - getResourceAsync, 80
  - getSyncChunk, 81
  - getSyncState, 81
  - noteStoreUrl, 82
  - setAuthData, 82
  - setNoteStoreUrl, 82
  - stop, 82
  - updateNote, 82
  - updateNotebook, 83
  - updateSavedSearch, 83
  - updateTag, 84
- quentier::INoteStoreDataElement, 84
- quentier::IQuentierException, 85
  - print, 87
- quentier::ISyncStateStorage, 89
  - notifySyncStateUpdated, 90
- quentier::ISyncStateStorage::ISyncState, 87
  - print, 88
- quentier::IUserStore, 90
  - checkVersion, 91
  - getAccountLimits, 91
  - getUser, 92
  - setAuthData, 92
- quentier::LimitedStack< T >, 92
- quentier::LinkedNotebook, 94
  - checkParameters, 95
  - clear, 96
  - guid, 96
  - hasGuid, 96
  - hasUpdateSequenceNumber, 96
  - print, 96
  - setGuid, 96
  - setUpdateSequenceNumber, 97
  - updateSequenceNumber, 97
- quentier::LocalStorageCacheManager, 97
  - print, 99
- quentier::LocalStorageCacheManagerException, 99
  - exceptionDisplayName, 100
- quentier::LocalStorageManager, 101
  - accountHighUsn, 109
  - addEnResource, 110
  - addLinkedNotebook, 110
  - addNote, 111
  - addNotebook, 111
  - addSavedSearch, 112
  - addTag, 112
  - addUser, 112
  - ClearDatabase, 108
  - deleteUser, 113
  - enResourceCount, 113
  - expungeEnResource, 114
  - expungeLinkedNotebook, 114
  - expungeNote, 115
  - expungeNotebook, 115
  - expungeNotelessTagsFromLinkedNotebooks, 116
  - expungeSavedSearch, 116
  - expungeTag, 116
  - expungeUser, 117
  - findDefaultNotebook, 117
  - findDefaultOrLastUsedNotebook, 118
  - findEnResource, 118
  - findLastUsedNotebook, 119
  - findLinkedNotebook, 119
  - findNote, 119
  - findNotebook, 120
  - findNoteLocalUidsWithSearchQuery, 120
  - findNotesWithSearchQuery, 121
  - findSavedSearch, 121
  - findTag, 122
  - findUser, 122
  - GetNoteOption, 107
  - GetResourceOption, 107
  - highestSupportedLocalStorageVersion, 123
  - isLocalStorageVersionTooHigh, 123
  - linkedNotebookCount, 123
  - listAllLinkedNotebooks, 124
  - listAllNotebooks, 124
  - listAllSavedSearches, 125
  - listAllSharedNotebooks, 126
  - listAllTags, 126
  - listAllTagsPerNote, 127
  - listLinkedNotebooks, 127
  - listNotebooks, 128
  - listNotes, 129
  - listNotesByLocalUids, 129
  - listNotesPerNotebook, 130



- listNotesPerNotebooksAndTags, 131
- listNotesPerTag, 131
- ListObjectsOption, 108
- listSavedSearches, 132
- listSharedNotebooksPerNotebookGuid, 133
- listTags, 133
- listTagsWithNoteLocalUids, 134
- LocalStorageManager, 109
- localStorageRequiresUpgrade, 135
- localStorageVersion, 135
- notebookCount, 136
- noteCount, 136
- noteCountPerNotebook, 136
- noteCountPerNotebooksAndTags, 137
- noteCountPerTag, 137
- noteCountsPerAllTags, 138
- OverrideLock, 108
- requiredLocalStoragePatches, 138
- savedSearchCount, 138
- StartupOption, 108
- switchUser, 139
- tagCount, 139
- updateEnResource, 140
- updateLinkedNotebook, 140
- updateNote, 140
- updateNotebook, 141
- UpdateNoteOption, 108
- UpdateResourceBinaryData, 109
- UpdateResourceMetadata, 109
- updateSavedSearch, 142
- updateTag, 142
- UpdateTags, 109
- updateUser, 143
- upgradeProgress, 143
- userCount, 144
- WithBinaryData, 108
- WithResourceBinaryData, 107
- WithResourceMetadata, 107
- quentier::LocalStorageManagerAsync, 144
- quentier::LoggerInitializationException, 151
  - exceptionDisplayName, 152
- quentier::LRUCache< Key, Value, Allocator >, 152
- quentier::Note, 153
  - checkParameters, 156
  - clear, 156
  - guid, 156
  - hasGuid, 156
  - hasUpdateSequenceNumber, 157
  - print, 157
  - setGuid, 157
  - setUpdateSequenceNumber, 157
  - updateSequenceNumber, 157
- quentier::Notebook, 158
  - checkParameters, 162
  - clear, 162
  - guid, 162
  - hasGuid, 162
  - hasUpdateSequenceNumber, 162
  - print, 162
  - setGuid, 163
  - setUpdateSequenceNumber, 163
  - updateSequenceNumber, 163
- quentier::NoteEditor, 164
  - backend, 167
  - clear, 167
  - convertToNote, 168
  - currentNoteLocalUid, 168
  - defaultFont, 168
  - defaultPalette, 168
  - idleTime, 168
  - inAppNoteLinkPasteRequested, 168
  - initialize, 169
  - isEditorPageModified, 169
  - isModified, 169
  - isNoteLoaded, 170
  - saveNoteToLocalStorage, 170
  - setAccount, 170
  - setBackend, 170
  - setCurrentNoteLocalUid, 170
  - setDefaultFont, 171
  - setDefaultPalette, 171
  - setFocus, 171
  - setInitialPageHtml, 171
  - setNoteDeletedPageHtml, 172
  - setNoteLoadingPageHtml, 172
  - setNoteNotFoundPageHtml, 172
  - setNoteTitle, 172
  - setTagIds, 172
  - setUndoStack, 173
  - undoStack, 173
- quentier::NoteEditorInitializationException, 173
  - exceptionDisplayName, 174
- quentier::NoteEditorPluginInitializationException, 175
  - exceptionDisplayName, 176
- quentier::NoteSearchQuery, 176
  - notebookModifier, 178
  - print, 179
  - queryString, 179
- quentier::NullPtrException, 179
  - exceptionDisplayName, 180
- quentier::Printable, 181
  - print, 182
- quentier::QuentierApplication, 182
- quentier::QuentierUndoCommand, 183
- quentier::Resource, 185
  - checkParameters, 187
  - clear, 187
  - guid, 187
  - hasGuid, 187
  - hasUpdateSequenceNumber, 188
  - print, 188
  - setGuid, 188
  - setUpdateSequenceNumber, 188
  - updateSequenceNumber, 188
- quentier::ResourceRecognitionIndexItem, 189
  - print, 191

- quentier::ResourceRecognitionIndexItem::BarcodeItem, 27
- quentier::ResourceRecognitionIndexItem::ObjectItem, 180
- quentier::ResourceRecognitionIndexItem::ShapeItem, 197
- quentier::ResourceRecognitionIndexItem::TextItem, 223
- quentier::ResourceRecognitionIndices, 191
  - print, 192
- quentier::SavedSearch, 193
  - checkParameters, 195
  - clear, 195
  - guid, 195
  - hasGuid, 196
  - hasUpdateSequenceNumber, 196
  - print, 196
  - setGuid, 196
  - setUpdateSequenceNumber, 196
  - updateSequenceNumber, 196
- quentier::SharedNote, 197
  - print, 199
- quentier::SharedNotebook, 200
  - print, 202
- quentier::ShortcutManager, 202
  - defaultShortcut, 203, 204
  - shortcut, 204
  - userShortcut, 204, 205
- quentier::SpellChecker, 207
- quentier::StringUtils, 208
- quentier::StringUtils::StringFilterPredicate, 208
- quentier::SynchronizationManager, 209
  - active, 211
  - authenticate, 211
  - authenticateCurrentAccount, 212
  - authenticationFinished, 212
  - authenticationRevoked, 212
  - detectedConflictDuringLocalChangesSending, 213
  - downloadNoteThumbnailsOption, 213
  - failed, 213
  - finished, 213
  - linkedNotebooksNotesDownloadProgress, 214
  - linkedNotebooksResourcesDownloadProgress, 214
  - linkedNotebooksSyncChunksDownloaded, 214
  - linkedNotebookSyncChunksDownloadProgress, 214
  - notesDownloadProgress, 215
  - preparedDirtyObjectsForSending, 215
  - preparedLinkedNotebooksDirtyObjectsForSending, 215
  - rateLimitExceeded, 215
  - remoteToLocalSyncDone, 216
  - remoteToLocalSyncStopped, 216
  - resourcesDownloadProgress, 216
  - revokeAuthentication, 217
  - sendLocalChangesStopped, 217
  - setAccount, 217
  - setAccountDone, 217
  - setDownloadInkNoteImages, 217
  - setDownloadInkNoteImagesDone, 217
  - setDownloadNoteThumbnails, 218
  - setDownloadNoteThumbnailsDone, 218
  - setInkNoteImagesStoragePath, 218
  - setInkNoteImagesStoragePathDone, 218
  - started, 218
  - stop, 219
  - stopped, 219
  - syncChunksDownloaded, 219
  - syncChunksDownloadProgress, 219
  - SynchronizationManager, 210
  - synchronize, 219
  - willRepeatRemoteToLocalSyncAfterSendingChanges, 220
- quentier::SysInfo, 220
- quentier::Tag, 220
  - checkParameters, 222
  - clear, 222
  - guid, 222
  - hasGuid, 222
  - hasUpdateSequenceNumber, 222
  - print, 223
  - setGuid, 223
  - setUpdateSequenceNumber, 223
  - updateSequenceNumber, 223
- quentier::UidGenerator, 224
- quentier::User, 224
  - print, 226
- QuentierApplication.h, 318
- QuentierCheckPtr.h, 318
- QuentierLogger.h, 259
- QuentierUndoCommand.h, 319
- queryString
  - quentier::NoteSearchQuery, 179
- rateLimitExceeded
  - quentier::SynchronizationManager, 215
- readFileRequestProcessed
  - quentier::FileIOProcessorAsync, 50
- readPasswordJobFinished
  - quentier::IKeychainService, 58
- RegisterMetatypes.h, 288
- remoteToLocalSyncDone
  - quentier::SynchronizationManager, 216
- remoteToLocalSyncStopped
  - quentier::SynchronizationManager, 216
- remove
  - quentier::ApplicationSettings, 20, 21
- removeLocalStorageBackup
  - quentier::ILocalStoragePatch, 68
- requiredLocalStoragePatches
  - quentier::LocalStorageManager, 138
- Resource.h, 288
- ResourceRecognitionIndexItem.h, 290
- ResourceRecognitionIndices.h, 292
- resourcesDownloadProgress
  - quentier::SynchronizationManager, 216
- restoreBackupProgress



- quentier::ILocalStoragePatch, 68
- restoreLocalStorageFromBackup
  - quentier::ILocalStoragePatch, 68
- revokeAuthentication
  - quentier::SynchronizationManager, 217
- SavedSearch.h, 293
- savedSearchCount
  - quentier::LocalStorageManager, 138
- saveNoteToLocalStorage
  - quentier::NoteEditor, 170
- sendLocalChangesStopped
  - quentier::SynchronizationManager, 217
- setAccount
  - quentier::NoteEditor, 170
  - quentier::SynchronizationManager, 217
- setAccountDone
  - quentier::SynchronizationManager, 217
- setAuthData
  - quentier::INoteStore, 82
  - quentier::IUserStore, 92
- setBackend
  - quentier::NoteEditor, 170
- setCurrentNoteLocalUid
  - quentier::NoteEditor, 170
- setDefaultFont
  - quentier::NoteEditor, 171
- setDefaultPalette
  - quentier::NoteEditor, 171
- setDisplayName
  - quentier::Account, 14
- setDownloadInkNoteImages
  - quentier::SynchronizationManager, 217
- setDownloadInkNoteImagesDone
  - quentier::SynchronizationManager, 217
- setDownloadNoteThumbnails
  - quentier::SynchronizationManager, 218
- setDownloadNoteThumbnailsDone
  - quentier::SynchronizationManager, 218
- setFocus
  - quentier::NoteEditor, 171
- setGuid
  - quentier::LinkedNotebook, 96
  - quentier::Note, 157
  - quentier::Notebook, 163
  - quentier::Resource, 188
  - quentier::SavedSearch, 196
  - quentier::Tag, 223
- setIdleTimePeriod
  - quentier::FileIOProcessorAsync, 50
- setInitialPageHtml
  - quentier::NoteEditor, 171
- setInkNoteImagesStoragePath
  - quentier::SynchronizationManager, 218
- setInkNoteImagesStoragePathDone
  - quentier::SynchronizationManager, 218
- setNoteDeletedPageHtml
  - quentier::NoteEditor, 172
- setNoteLoadingPageHtml
  - quentier::NoteEditor, 172
- setNoteNotFoundPageHtml
  - quentier::NoteEditor, 172
- setNoteStoreUrl
  - quentier::INoteStore, 82
- setNoteTitle
  - quentier::NoteEditor, 172
- setTagIds
  - quentier::NoteEditor, 172
- setUndoStack
  - quentier::NoteEditor, 173
- setUpdateSequenceNumber
  - quentier::LinkedNotebook, 97
  - quentier::Note, 157
  - quentier::Notebook, 163
  - quentier::Resource, 188
  - quentier::SavedSearch, 196
  - quentier::Tag, 223
- setValue
  - quentier::ApplicationSettings, 21
- shardId
  - quentier::Account, 14
- SharedNote.h, 294
- SharedNotebook.h, 296
- shortcut
  - quentier::ShortcutManager, 204
- ShortcutManager.h, 319
- Size.h, 322
- SpellChecker.h, 266
- StandardPaths.h, 322
- startDeletePasswordJob
  - quentier::IKeychainService, 59
- started
  - quentier::SynchronizationManager, 218
- startReadPasswordJob
  - quentier::IKeychainService, 59
- StartupOption
  - quentier::LocalStorageManager, 108
- startWritePasswordJob
  - quentier::IKeychainService, 59
- stop
  - quentier::INoteStore, 82
  - quentier::SynchronizationManager, 219
- stopped
  - quentier::SynchronizationManager, 219
- StringUtils.h, 323
- SuppressWarnings.h, 323
- switchUser
  - quentier::LocalStorageManager, 139
- syncChunksDownloaded
  - quentier::SynchronizationManager, 219
- syncChunksDownloadProgress
  - quentier::SynchronizationManager, 219
- SynchronizationManager
  - quentier::SynchronizationManager, 210
- SynchronizationManager.h, 273
- synchronize
  - quentier::SynchronizationManager, 219

- SysInfo.h, [325](#)
  - System.h, [325](#)
- Tag.h, [298](#)
- tagCount
  - quentier::LocalStorageManager, [139](#)
- TagSortByParentChildRelations.h, [326](#)
- toVersion
  - quentier::ILocalStoragePatch, [69](#)
- type
  - quentier::Account, [14](#)
- UidGenerator.h, [326](#)
- undoStack
  - quentier::NoteEditor, [173](#)
- updateEnResource
  - quentier::LocalStorageManager, [140](#)
- updateLinkedNotebook
  - quentier::LocalStorageManager, [140](#)
- updateNote
  - quentier::INoteStore, [82](#)
  - quentier::LocalStorageManager, [140](#)
- updateNotebook
  - quentier::INoteStore, [83](#)
  - quentier::LocalStorageManager, [141](#)
- UpdateNoteOption
  - quentier::LocalStorageManager, [108](#)
- UpdateResourceBinaryData
  - quentier::LocalStorageManager, [109](#)
- UpdateResourceMetadata
  - quentier::LocalStorageManager, [109](#)
- updateSavedSearch
  - quentier::INoteStore, [83](#)
  - quentier::LocalStorageManager, [142](#)
- updateSequenceNumber
  - quentier::LinkedNotebook, [97](#)
  - quentier::Note, [157](#)
  - quentier::Notebook, [163](#)
  - quentier::Resource, [188](#)
  - quentier::SavedSearch, [196](#)
  - quentier::Tag, [223](#)
- updateTag
  - quentier::INoteStore, [84](#)
  - quentier::LocalStorageManager, [142](#)
- UpdateTags
  - quentier::LocalStorageManager, [109](#)
- updateUser
  - quentier::LocalStorageManager, [143](#)
- upgradeProgress
  - quentier::LocalStorageManager, [143](#)
- User.h, [299](#)
- userCount
  - quentier::LocalStorageManager, [144](#)
- userShortcut
  - quentier::ShortcutManager, [204](#), [205](#)
- value
  - quentier::ApplicationSettings, [22](#)
- willRepeatRemoteToLocalSyncAfterSendingChanges
  - quentier::SynchronizationManager, [220](#)
- WithBinaryData
  - quentier::LocalStorageManager, [108](#)
- WithResourceBinaryData
  - quentier::LocalStorageManager, [107](#)
- WithResourceMetadata
  - quentier::LocalStorageManager, [107](#)
- writeFileRequestProcessed
  - quentier::FileIOProcessorAsync, [51](#)
- writePasswordJobFinished
  - quentier::IKeychainService, [60](#)