

SDK

2.1.4 [async]

Generated by Doxygen 1.9.1

1 API for C++	1
2 Class diagrams	3
2.1 Overview	3
2.2 CloudBackend - core object	4
2.3 Object	5
2.4 Group	6
3 Sync vs. Async	7
3.1 Choosing a Synchronous or Asynchronous coding style	7
4 Sequence diagrams - async	11
4.1 Login asynchronous UML	11
4.2 Query asynchronous UML	13
5 Example code - combined lists grouped by call	15
5.1 login	15
5.2 upload	15
5.3 getStreams	15
5.4 uploadStream	15
5.5 downloadStream	15
5.6 query - select	15
5.7 databases - tenant	15
5.8 Complete program	15
6 Full example - async	17
7 Document list	21
8 Namespace Index	23
8.1 Namespace List	23
9 Hierarchical Index	25
9.1 Class Hierarchy	25
10 Class Index	29
10.1 Class List	29
11 Namespace Documentation	33
11.1 cbe Namespace Reference	33
11.1.1 Detailed Description	36
11.1.2 Typedef Documentation	36
11.1.2.1 AclGroupId	36
11.1.2.2 AclMap	36
11.1.2.3 Date	36
11.1.2.4 ErrorCode	36

11.1.2.5 KeyValues	37
11.1.2.6 MemberBanInfo	37
11.1.3 Enumeration Type Documentation	37
11.1.3.1 AccountStatus	37
11.1.3.2 AclScope	37
11.1.3.3 DefaultCtor	38
11.1.3.4 FilterOrder	38
11.1.3.5 ItemType	38
11.1.3.6 ObjectType	38
11.1.3.7 Permissions	39
11.1.3.8 PublishAccess	39
11.1.3.9 PublishVisibility	40
11.1.3.10 Visibility	40
11.2 cbe::delegate Namespace Reference	40
11.2.1 Detailed Description	42
11.2.2 Typedef Documentation	42
11.2.2.1 AclDelegatePtr	43
11.2.2.2 AddRoleMemberDelegatePtr	43
11.2.2.3 BanDelegatePtr	43
11.2.2.4 CloudBackendListenerDelegatePtr	43
11.2.2.5 CreateAccountDelegatePtr	43
11.2.2.6 CreateContainerDelegatePtr	43
11.2.2.7 CreateGroupDelegatePtr	43
11.2.2.8 CreateObjectDelegatePtr	43
11.2.2.9 CreateRoleDelegatePtr	44
11.2.2.10 DownloadBinaryDelegatePtr	44
11.2.2.11 DownloadDelegatePtr	44
11.2.2.12 GetStreamsDelegatePtr	44
11.2.2.13 GetSubscriptionsDelegatePtr	44
11.2.2.14 ICloudBackendListenerPtr	44
11.2.2.15 JoinDelegatePtr	44
11.2.2.16 KickDelegatePtr	44
11.2.2.17 LeaveDelegatePtr	45
11.2.2.18 ListGroupsDelegatePtr	45
11.2.2.19 ListMembersDelegatePtr	45
11.2.2.20 ListRolesDelegatePtr	45
11.2.2.21 ListSharesDelegatePtr	45
11.2.2.22 LoginDelegatePtr	45
11.2.2.23 ProgressEventFn	45
11.2.2.24 PublishDelegatePtr	45
11.2.2.25 QueryDelegatePtr	46
11.2.2.26 QueryJoinDelegatePtr	46

11.2.2.27 RemoveRoleDelegatePtr	46
11.2.2.28 RemoveRoleMemberDelegatePtr	46
11.2.2.29 SearchGroupsDelegatePtr	46
11.2.2.30 ShareDelegatePtr	46
11.2.2.31 SubscribeDelegatePtr	46
11.2.2.32 UnBanDelegatePtr	47
11.2.2.33 UnPublishDelegatePtr	47
11.2.2.34 UnShareDelegatePtr	47
11.2.2.35 UnSubscribeDelegatePtr	47
11.2.2.36 UpdateKeyValuesDelegatePtr	47
11.2.2.37 UploadDelegatePtr	47
11.3 cbe::delegate::container Namespace Reference	47
11.3.1 Detailed Description	48
11.3.2 Typedef Documentation	48
11.3.2.1 MoveDelegatePtr	48
11.3.2.2 RemoveDelegatePtr	48
11.3.2.3 RenameDelegatePtr	48
11.4 cbe::delegate::group Namespace Reference	48
11.4.1 Detailed Description	49
11.4.2 Typedef Documentation	49
11.4.2.1 JoinDelegatePtr	49
11.4.2.2 RemoveDelegatePtr	49
11.4.2.3 RenameDelegatePtr	49
11.5 cbe::delegate::object Namespace Reference	49
11.5.1 Detailed Description	49
11.5.2 Typedef Documentation	50
11.5.2.1 MoveDelegatePtr	50
11.5.2.2 RemoveDelegatePtr	50
11.5.2.3 RenameDelegatePtr	50
11.6 cbe::util Namespace Reference	50
11.6.1 Detailed Description	50
12 Class Documentation	51
12.1 cbe::Account Class Reference	51
12.1.1 Detailed Description	52
12.1.2 Constructor & Destructor Documentation	52
12.1.2.1 Account()	52
12.1.3 Member Function Documentation	52
12.1.3.1 client()	52
12.1.3.2 databases()	52
12.1.3.3 firstName()	52
12.1.3.4 libContainerId()	52

12.1.3.5 operator bool()	52
12.1.3.6 password()	53
12.1.3.7 rootContainer()	53
12.1.3.8 rootDatabase()	53
12.1.3.9 source()	53
12.1.3.10 tenantContainerId()	53
12.1.3.11 username()	53
12.2 cbe::delegate::AclDelegate Class Reference	53
12.2.1 Detailed Description	54
12.2.2 Member Function Documentation	54
12.2.2.1 onAclError()	54
12.2.2.2 onAclSuccess()	54
12.3 cbe::delegate::AddRoleMemberDelegate Class Reference	54
12.3.1 Detailed Description	55
12.3.2 Member Function Documentation	55
12.3.2.1 onAddRoleMemberError()	55
12.3.2.2 onAddRoleMemberSuccess()	55
12.4 cbe::util::Optional< T >::BadAccess Class Reference	55
12.4.1 Detailed Description	56
12.5 cbe::delegate::BanDelegate Class Reference	56
12.5.1 Detailed Description	56
12.5.2 Member Function Documentation	56
12.5.2.1 onBanError()	57
12.5.2.2 onBanSuccess()	57
12.6 cbe::delegate::BanSuccess Class Reference	57
12.6.1 Detailed Description	57
12.7 cbe::delegate::ChunkTransferred Class Reference	57
12.7.1 Detailed Description	58
12.8 cbe::CloudBackend Class Reference	58
12.8.1 Detailed Description	60
12.8.2 Member Typedef Documentation	60
12.8.2.1 CloudBackendListenerDelegatePtr	60
12.8.2.2 CreateAccountDelegatePtr	60
12.8.2.3 LoginDelegatePtr	60
12.8.2.4 QueryDelegatePtr	60
12.8.2.5 QueryJoinDelegatePtr	61
12.8.3 Constructor & Destructor Documentation	61
12.8.3.1 CloudBackend()	61
12.8.4 Member Function Documentation	61
12.8.4.1 account()	61
12.8.4.2 addListener()	61
12.8.4.3 castContainer()	61

12.8.4.4	castObject()	62
12.8.4.5	clearCache()	62
12.8.4.6	createAccount()	62
12.8.4.7	groupManager()	62
12.8.4.8	login()	63
12.8.4.9	operator bool()	64
12.8.4.10	publishManager()	65
12.8.4.11	query() [1/4]	65
12.8.4.12	query() [2/4]	65
12.8.4.13	query() [3/4]	65
12.8.4.14	query() [4/4]	66
12.8.4.15	queryWithPath() [1/2]	67
12.8.4.16	queryWithPath() [2/2]	67
12.8.4.17	removeListener()	67
12.8.4.18	search() [1/2]	67
12.8.4.19	search() [2/2]	68
12.8.4.20	shareManager()	68
12.8.4.21	subscribeManager()	68
12.8.4.22	terminate()	69
12.8.4.23	version()	69
12.9	cbe::delegate::CloudBackendListenerDelegate Class Reference	69
12.9.1	Member Function Documentation	70
12.9.1.1	onRemoteContainerAdded()	70
12.9.1.2	onRemoteContainerMoved()	70
12.9.1.3	onRemoteContainerRemoved()	70
12.9.1.4	onRemoteContainerRenamed()	71
12.9.1.5	onRemoteObjectAdded()	71
12.9.1.6	onRemoteObjectMoved()	71
12.9.1.7	onRemoteObjectRemoved()	71
12.9.1.8	onRemoteObjectRenamed()	71
12.10	cbe::Container Class Reference	72
12.10.1	Detailed Description	74
12.10.2	Member Typedef Documentation	74
12.10.2.1	CreateContainerDelegatePtr	74
12.10.2.2	CreateObjectDelegatePtr	74
12.10.2.3	GetAclDelegatePtr	74
12.10.2.4	MoveDelegatePtr	74
12.10.2.5	PublishDelegatePtr	74
12.10.2.6	QueryDelegatePtr	75
12.10.2.7	QueryJoinDelegatePtr	75
12.10.2.8	RemoveDelegatePtr	75
12.10.2.9	RenameDelegatePtr	75

12.10.2.10 SetAclDelegatePtr	75
12.10.2.11 ShareDelegatePtr	75
12.10.2.12 UnPublishDelegatePtr	75
12.10.2.13 UnShareDelegatePtr	75
12.10.2.14 UnSubscribeDelegatePtr	75
12.10.2.15 UploadDelegatePtr	75
12.10.3 Member Function Documentation	75
12.10.3.1 createContainer()	76
12.10.3.2 createObject() [1/2]	76
12.10.3.3 createObject() [2/2]	76
12.10.3.4 getAcl()	76
12.10.3.5 move()	77
12.10.3.6 publish()	77
12.10.3.7 query() [1/4]	77
12.10.3.8 query() [2/4]	77
12.10.3.9 query() [3/4]	78
12.10.3.10 query() [4/4]	78
12.10.3.11 queryWithPath()	78
12.10.3.12 remove()	79
12.10.3.13 rename()	79
12.10.3.14 search() [1/2]	79
12.10.3.15 search() [2/2]	79
12.10.3.16 setAcl()	80
12.10.3.17 share()	80
12.10.3.18 unPublish()	80
12.10.3.19 unShare()	81
12.10.3.20 unSubscribe()	81
12.10.3.21 upload() [1/3]	81
12.10.3.22 upload() [2/3]	82
12.10.3.23 upload() [3/3]	83
12.11 cbe::util::Context Struct Reference	83
12.12 cbe::delegate::CreateAccountDelegate Class Reference	83
12.12.1 Detailed Description	84
12.12.2 Member Function Documentation	84
12.12.2.1 onCreateAccountError()	84
12.12.2.2 onCreateAccountSuccess()	84
12.13 cbe::delegate::CreateContainerDelegate Class Reference	84
12.13.1 Detailed Description	85
12.13.2 Member Function Documentation	85
12.13.2.1 onCreateContainerError()	85
12.13.2.2 onCreateContainerSuccess()	85
12.14 cbe::delegate::CreateGroupDelegate Class Reference	85

12.14.1 Detailed Description	85
12.14.2 Member Function Documentation	86
12.14.2.1 onCreateGroupError()	86
12.14.2.2 onCreateGroupSuccess()	86
12.15 cbe::delegate::CreateObjectDelegate Class Reference	86
12.15.1 Detailed Description	86
12.15.2 Member Function Documentation	86
12.15.2.1 onCreateObjectError()	86
12.15.2.2 onCreateObjectSuccess()	87
12.16 cbe::delegate::CreateRoleDelegate Class Reference	87
12.16.1 Detailed Description	87
12.16.2 Member Function Documentation	87
12.16.2.1 onCreateRoleError()	87
12.16.2.2 onCreateRoleSuccess()	87
12.17 cbe::Database Class Reference	88
12.17.1 Detailed Description	88
12.17.2 Member Function Documentation	88
12.17.2.1 containerId()	88
12.17.2.2 created()	89
12.17.2.3 databaseId()	89
12.17.2.4 name()	89
12.17.2.5 ownerId()	89
12.17.2.6 readLocked()	89
12.17.2.7 rootContainer()	89
12.17.2.8 writeLocked()	90
12.18 cbe::delegate::DownloadBinaryDelegate Class Reference	90
12.18.1 Detailed Description	90
12.18.2 Member Function Documentation	90
12.18.2.1 onChunkReceived()	90
12.18.2.2 onDownloadBinaryError()	91
12.18.2.3 onDownloadBinarySuccess()	91
12.19 cbe::delegate::DownloadBinarySuccess Class Reference	91
12.19.1 Detailed Description	92
12.19.2 Member Function Documentation	92
12.19.2.1 operator bool()	92
12.20 cbe::delegate::DownloadDelegate Class Reference	92
12.20.1 Detailed Description	92
12.20.2 Member Function Documentation	92
12.20.2.1 onChunkReceived()	93
12.20.2.2 onDownloadError()	93
12.20.2.3 onDownloadSuccess()	93
12.21 cbe::delegate::DownloadSuccess Class Reference	93

12.21.1 Detailed Description	94
12.21.2 Member Function Documentation	94
12.21.2.1 operator bool()	94
12.22 cbe::delegate::Error Class Reference	95
12.22.1 Detailed Description	95
12.22.2 Member Function Documentation	95
12.22.2.1 operator bool()	95
12.22.3 Friends And Related Function Documentation	95
12.22.3.1 operator<<	95
12.22.4 Member Data Documentation	96
12.22.4.1 errorCode	96
12.22.4.2 message	96
12.22.4.3 reason	96
12.23 cbe::delegate::ItemError::Error Struct Reference	96
12.24 cbe::delegate::JoinError::Error Class Reference	97
12.25 cbe::delegate::LoadError::Error Class Reference	97
12.25.1 Member Data Documentation	97
12.25.1.1 errorCode	98
12.26 cbe::delegate::AclDelegate::ErrorInfo Struct Reference	98
12.26.1 Detailed Description	99
12.27 cbe::delegate::AddRoleMemberDelegate::ErrorInfo Struct Reference	99
12.27.1 Detailed Description	100
12.28 cbe::delegate::BanDelegate::ErrorInfo Struct Reference	100
12.28.1 Detailed Description	101
12.29 cbe::delegate::container::MoveDelegate::ErrorInfo Struct Reference	102
12.29.1 Detailed Description	102
12.30 cbe::delegate::container::RemoveDelegate::ErrorInfo Struct Reference	103
12.30.1 Detailed Description	103
12.31 cbe::delegate::container::RenameDelegate::ErrorInfo Struct Reference	104
12.31.1 Detailed Description	104
12.32 cbe::delegate::CreateAccountDelegate::ErrorInfo Struct Reference	105
12.32.1 Detailed Description	105
12.33 cbe::delegate::CreateContainerDelegate::ErrorInfo Struct Reference	106
12.33.1 Detailed Description	106
12.34 cbe::delegate::CreateGroupDelegate::ErrorInfo Struct Reference	107
12.34.1 Detailed Description	107
12.35 cbe::delegate::CreateObjectDelegate::ErrorInfo Struct Reference	108
12.35.1 Detailed Description	108
12.36 cbe::delegate::CreateRoleDelegate::ErrorInfo Struct Reference	109
12.36.1 Detailed Description	109
12.37 cbe::delegate::DownloadBinaryDelegate::ErrorInfo Struct Reference	110
12.37.1 Detailed Description	110

12.38 cbe::delegate::DownloadDelegate::ErrorInfo Struct Reference	111
12.38.1 Detailed Description	111
12.39 cbe::delegate::GetStreamsDelegate::ErrorInfo Struct Reference	112
12.39.1 Detailed Description	112
12.40 cbe::delegate::GetSubscriptionsDelegate::ErrorInfo Struct Reference	113
12.40.1 Detailed Description	113
12.41 cbe::delegate::group::JoinDelegate::ErrorInfo Struct Reference	114
12.41.1 Detailed Description	114
12.42 cbe::delegate::group::RemoveDelegate::ErrorInfo Struct Reference	115
12.42.1 Detailed Description	115
12.43 cbe::delegate::group::RenameDelegate::ErrorInfo Struct Reference	116
12.43.1 Detailed Description	116
12.44 cbe::delegate::KickDelegate::ErrorInfo Struct Reference	117
12.44.1 Detailed Description	117
12.45 cbe::delegate::LeaveDelegate::ErrorInfo Struct Reference	118
12.45.1 Detailed Description	118
12.46 cbe::delegate::ListGroupDelegate::ErrorInfo Struct Reference	119
12.46.1 Detailed Description	119
12.47 cbe::delegate::ListMembersDelegate::ErrorInfo Struct Reference	120
12.47.1 Detailed Description	120
12.48 cbe::delegate::ListRolesDelegate::ErrorInfo Struct Reference	121
12.48.1 Detailed Description	121
12.49 cbe::delegate::ListSharesDelegate::ErrorInfo Struct Reference	122
12.49.1 Detailed Description	122
12.50 cbe::delegate::LoginDelegate::ErrorInfo Struct Reference	123
12.50.1 Detailed Description	123
12.51 cbe::delegate::object::MoveDelegate::ErrorInfo Struct Reference	124
12.51.1 Detailed Description	124
12.52 cbe::delegate::object::RemoveDelegate::ErrorInfo Struct Reference	125
12.52.1 Detailed Description	125
12.53 cbe::delegate::object::RenameDelegate::ErrorInfo Struct Reference	126
12.53.1 Detailed Description	126
12.54 cbe::delegate::PublishDelegate::ErrorInfo Struct Reference	127
12.54.1 Detailed Description	127
12.55 cbe::delegate::QueryDelegate::ErrorInfo Struct Reference	128
12.55.1 Detailed Description	128
12.56 cbe::delegate::RemoveRoleDelegate::ErrorInfo Struct Reference	129
12.56.1 Detailed Description	129
12.57 cbe::delegate::RemoveRoleMemberDelegate::ErrorInfo Struct Reference	130
12.57.1 Detailed Description	130
12.58 cbe::delegate::SearchGroupsDelegate::ErrorInfo Struct Reference	131
12.58.1 Detailed Description	131

12.59 cbe::delegate::ShareDelegate::ErrorInfo Struct Reference	132
12.59.1 Detailed Description	132
12.60 cbe::delegate::SubscribeDelegate::ErrorInfo Struct Reference	133
12.60.1 Detailed Description	133
12.61 cbe::delegate::UnBanDelegate::ErrorInfo Struct Reference	134
12.61.1 Detailed Description	134
12.62 cbe::delegate::UnPublishDelegate::ErrorInfo Struct Reference	135
12.62.1 Detailed Description	135
12.63 cbe::delegate::UnShareDelegate::ErrorInfo Struct Reference	136
12.63.1 Detailed Description	136
12.64 cbe::delegate::UnSubscribeDelegate::ErrorInfo Struct Reference	137
12.64.1 Detailed Description	137
12.65 cbe::delegate::UpdateKeyValuesDelegate::ErrorInfo Struct Reference	138
12.65.1 Detailed Description	138
12.66 cbe::delegate::UploadDelegate::ErrorInfo Struct Reference	139
12.66.1 Detailed Description	139
12.67 cbe::util::ErrorInfo Struct Reference	140
12.68 cbe::util::ErrorInfoImpl< ErrorT > Struct Template Reference	141
12.69 cbe::QueryChain::Exception Struct Reference	142
12.70 cbe::util::Exception Struct Reference	143
12.70.1 Detailed Description	143
12.70.2 Member Function Documentation	144
12.70.2.1 derivedCbeException()	144
12.71 cbe::util::ExceptionImpl< ErrorInfoT > Struct Template Reference	144
12.71.1 Detailed Description	145
12.72 cbe::Filter Class Reference	145
12.72.1 Detailed Description	147
12.72.2 Member Function Documentation	147
12.72.2.1 getDataType()	147
12.72.2.2 getItemOrder()	147
12.72.2.3 getOffset()	147
12.72.2.4 getQuery()	147
12.72.2.5 setAscending()	147
12.72.2.6 setByPassCache()	148
12.72.2.7 setContainerFirst()	148
12.72.2.8 setCount()	148
12.72.2.9 setDataType()	148
12.72.2.10 setDeleted()	149
12.72.2.11 setItemOrder()	149
12.72.2.12 setOffset()	149
12.72.2.13 setQuery()	149
12.73 cbe::delegate::GetStreamsDelegate Class Reference	150

12.73.1 Detailed Description	150
12.73.2 Member Function Documentation	150
12.73.2.1 onGetStreamsError()	150
12.73.2.2 onGetStreamsSuccess()	150
12.74 cbe::delegate::GetSubscriptionsDelegate Class Reference	151
12.74.1 Detailed Description	151
12.74.2 Member Function Documentation	151
12.74.2.1 onGetSubscriptionsError()	151
12.74.2.2 onGetSubscriptionsSuccess()	151
12.75 cbe::Group Class Reference	151
12.75.1 Detailed Description	153
12.75.2 Member Typedef Documentation	153
12.75.2.1 CreateGroupDelegatePtr	153
12.75.2.2 CreateRoleDelegatePtr	153
12.75.2.3 JoinDelegatePtr	153
12.75.2.4 LeaveDelegatePtr	153
12.75.2.5 ListBannedMembersDelegatePtr	153
12.75.2.6 ListMembersDelegatePtr	153
12.75.2.7 ListRolesDelegatePtr	153
12.75.2.8 RemoveDelegatePtr	153
12.75.2.9 RemoveRoleDelegatePtr	153
12.75.2.10 RenameDelegatePtr	153
12.75.3 Constructor & Destructor Documentation	154
12.75.3.1 Group()	154
12.75.4 Member Function Documentation	154
12.75.4.1 createGroup()	154
12.75.4.2 createRole()	154
12.75.4.3 getVisibility()	155
12.75.4.4 groupContainer()	155
12.75.4.5 id()	155
12.75.4.6 join()	155
12.75.4.7 joined()	155
12.75.4.8 leave()	155
12.75.4.9 listBannedMembers()	156
12.75.4.10 listMembers()	156
12.75.4.11 listRoles()	156
12.75.4.12 name()	156
12.75.4.13 operator bool()	156
12.75.4.14 parentId()	157
12.75.4.15 remove()	157
12.75.4.16 removeRole()	157
12.75.4.17 rename()	157

12.75.4.18 requests()	157
12.76 cbe::GroupFilter Class Reference	158
12.76.1 Detailed Description	158
12.76.2 Member Function Documentation	158
12.76.2.1 getAscending()	158
12.76.2.2 getCount()	158
12.76.2.3 getDeleted()	159
12.76.2.4 getFilter()	159
12.76.2.5 getGroupOrder()	159
12.76.2.6 getOffset()	159
12.76.2.7 getOrder()	159
12.76.2.8 getPublicFirst()	159
12.76.2.9 getQuery()	159
12.76.2.10 setAscending()	159
12.76.2.11 setCount()	159
12.76.2.12 setDeleted()	159
12.76.2.13 setFilter()	159
12.76.2.14 setOffset()	160
12.76.2.15 setQuery()	160
12.76.3 Friends And Related Function Documentation	160
12.76.3.1 operator<<	160
12.77 cbe::GroupManager Class Reference	160
12.77.1 Detailed Description	160
12.77.2 Member Typedef Documentation	161
12.77.2.1 ListGroupsDelegatePtr	161
12.77.2.2 SearchGroupsDelegatePtr	161
12.77.3 Constructor & Destructor Documentation	161
12.77.3.1 GroupManager()	161
12.77.4 Member Function Documentation	161
12.77.4.1 getTenantId()	161
12.77.4.2 listGroups()	161
12.77.4.3 operator bool()	161
12.77.4.4 searchGroups()	162
12.78 cbe::GroupQueryResult Class Reference	162
12.78.1 Detailed Description	162
12.78.2 Member Function Documentation	162
12.78.2.1 containsGroup()	163
12.78.2.2 filter()	163
12.78.2.3 getGroupsSnapshot()	163
12.78.2.4 GroupsLoaded()	163
12.78.2.5 totalCount()	163
12.79 cbe::delegate::ICloudBackendListener Class Reference	164

12.79.1 Member Function Documentation	164
12.79.1.1 onRemoteContainerAdded()	164
12.79.1.2 onRemoteContainerMoved()	164
12.79.1.3 onRemoteContainerRemoved()	165
12.79.1.4 onRemoteContainerRenamed()	165
12.79.1.5 onRemoteObjectAdded()	165
12.79.1.6 onRemoteObjectMoved()	165
12.79.1.7 onRemoteObjectRemoved()	166
12.79.1.8 onRemoteObjectRenamed()	166
12.80 cbe::Item Class Reference	166
12.80.1 Detailed Description	167
12.80.2 Member Function Documentation	167
12.80.2.1 aclMap()	167
12.80.2.2 aclTag()	168
12.80.2.3 created()	168
12.80.2.4 dataLoaded()	168
12.80.2.5 deleted()	168
12.80.2.6 description()	168
12.80.2.7 driveId()	168
12.80.2.8 getPublished()	168
12.80.2.9 getShareFromUserId()	168
12.80.2.10 getShareIds()	168
12.80.2.11 getSubscribe()	168
12.80.2.12 getUserFromShareId()	169
12.80.2.13 hasPublished()	169
12.80.2.14 hasSubscribe()	169
12.80.2.15 id()	169
12.80.2.16 idLoaded()	169
12.80.2.17 name()	169
12.80.2.18 oldParentId()	169
12.80.2.19 operator bool()	169
12.80.2.20 ownerId()	169
12.80.2.21 parentId()	169
12.80.2.22 path()	170
12.80.2.23 type()	170
12.80.2.24 updated()	170
12.80.2.25 username()	170
12.81 cbe::delegate::ItemError Class Reference	170
12.81.1 Detailed Description	170
12.81.2 Member Function Documentation	170
12.81.2.1 onItemError()	170
12.82 cbe::delegate::group::JoinDelegate Class Reference	171

12.82.1 Detailed Description	171
12.82.2 Member Function Documentation	171
12.82.2.1 onJoinError()	171
12.82.2.2 onJoinSuccess()	171
12.83 cbe::delegate::JoinDelegate Class Reference	171
12.83.1 Detailed Description	172
12.83.2 Member Typedef Documentation	172
12.83.2.1 ErrorInfo	172
12.83.3 Member Function Documentation	172
12.83.3.1 onJoinError()	172
12.83.3.2 onJoinSuccess()	172
12.84 cbe::delegate::JoinError Class Reference	172
12.84.1 Detailed Description	173
12.84.2 Member Function Documentation	173
12.84.2.1 onJoinError()	173
12.85 cbe::delegate::KickDelegate Class Reference	173
12.85.1 Detailed Description	173
12.85.2 Member Function Documentation	174
12.85.2.1 onKickError()	174
12.85.2.2 onKickSuccess()	174
12.86 cbe::delegate::KickSuccess Class Reference	174
12.86.1 Detailed Description	174
12.87 cbe::delegate::LeaveDelegate Class Reference	174
12.87.1 Detailed Description	175
12.87.2 Member Function Documentation	175
12.87.2.1 onLeaveError()	175
12.87.2.2 onLeaveSuccess()	175
12.88 cbe::delegate::LeaveSuccess Class Reference	175
12.88.1 Member Function Documentation	175
12.88.1.1 operator bool()	176
12.89 cbe::delegate::ListGroupDelegate Class Reference	176
12.89.1 Detailed Description	176
12.89.2 Member Function Documentation	176
12.89.2.1 onListGroupError()	176
12.89.2.2 onListGroupSuccess()	176
12.90 cbe::delegate::ListMembersDelegate Class Reference	177
12.90.1 Detailed Description	177
12.90.2 Member Function Documentation	177
12.90.2.1 onListMembersError()	177
12.90.2.2 onListMembersSuccess()	177
12.91 cbe::delegate::ListRolesDelegate Class Reference	177
12.91.1 Detailed Description	178

12.91.2 Member Function Documentation	178
12.91.2.1 onListRolesError()	178
12.91.2.2 onListRolesSuccess()	178
12.92 cbe::delegate::ListSharesDelegate Class Reference	178
12.92.1 Detailed Description	179
12.92.2 Member Function Documentation	179
12.92.2.1 onListSharesError()	179
12.92.2.2 onListSharesSuccess()	179
12.93 cbe::delegate::LoadError Class Reference	179
12.93.1 Detailed Description	179
12.93.2 Member Function Documentation	179
12.93.2.1 onLoadError()	180
12.94 cbe::delegate::LoginDelegate Class Reference	180
12.94.1 Detailed Description	180
12.94.2 Member Typedef Documentation	180
12.94.2.1 Success	180
12.94.3 Member Function Documentation	180
12.94.3.1 onLoginError()	181
12.94.3.2 onLoginSuccess()	181
12.95 cbe::MatchesID< IdT > Class Template Reference	181
12.95.1 Detailed Description	181
12.96 cbe::Member Class Reference	181
12.96.1 Detailed Description	182
12.96.2 Member Typedef Documentation	182
12.96.2.1 BanDelegatePtr	182
12.96.2.2 KickDelegatePtr	182
12.96.2.3 UnBanDelegatePtr	182
12.96.3 Constructor & Destructor Documentation	183
12.96.3.1 Member()	183
12.96.4 Member Function Documentation	183
12.96.4.1 ban()	183
12.96.4.2 groupId()	183
12.96.4.3 kick()	183
12.96.4.4 memberId()	183
12.96.4.5 name()	183
12.96.4.6 operator bool()	184
12.96.4.7 unBan()	184
12.96.4.8 visibility()	184
12.97 cbe::delegate::container::MoveDelegate Class Reference	184
12.97.1 Detailed Description	184
12.97.2 Member Function Documentation	185
12.97.2.1 onMoveError()	185

12.97.2.2 onMoveSuccess()	185
12.98 cbe::delegate::object::MoveDelegate Class Reference	185
12.98.1 Detailed Description	185
12.98.2 Member Function Documentation	185
12.98.2.1 onMoveError()	185
12.98.2.2 onMoveSuccess()	186
12.99 cbe::Object Class Reference	186
12.99.1 Detailed Description	188
12.99.2 Member Typedef Documentation	188
12.99.2.1 DownloadBinaryDelegatePtr	188
12.99.2.2 DownloadDelegatePtr	188
12.99.2.3 GetAclDelegatePtr	188
12.99.2.4 GetStreamsDelegatePtr	188
12.99.2.5 MoveDelegatePtr	189
12.99.2.6 PublishDelegatePtr	189
12.99.2.7 RemoveDelegatePtr	189
12.99.2.8 RenameDelegatePtr	189
12.99.2.9 SetAclDelegatePtr	189
12.99.2.10 ShareDelegatePtr	189
12.99.2.11 Streams	189
12.99.2.12 UnPublishDelegatePtr	189
12.99.2.13 UnShareDelegatePtr	189
12.99.2.14 UnSubscribeDelegatePtr	189
12.99.2.15 UpdateKeyValuesDelegatePtr	189
12.99.2.16 UploadDelegatePtr	190
12.99.3 Member Function Documentation	190
12.99.3.1 download() [1/2]	190
12.99.3.2 download() [2/2]	190
12.99.3.3 downloadStream()	190
12.99.3.4 getAcl()	191
12.99.3.5 getMimeType()	192
12.99.3.6 getObjectType()	192
12.99.3.7 getStreams()	192
12.99.3.8 move()	193
12.99.3.9 publish()	193
12.99.3.10 remove()	193
12.99.3.11 rename()	194
12.99.3.12 setAcl()	194
12.99.3.13 share()	194
12.99.3.14 unPublish()	195
12.99.3.15 unShare()	195
12.99.3.16 unSubscribe()	195

12.99.3.17 updateKeyValues() [1/2]	195
12.99.3.18 updateKeyValues() [2/2]	196
12.99.3.19 uploadStream()	196
12.100 cbe::util::Optional< T > Class Template Reference	197
12.100.1 Detailed Description	198
12.100.2 Member Function Documentation	198
12.100.2.1 operator bool()	198
12.101 cbe::Publish Class Reference	199
12.101.1 Detailed Description	199
12.101.2 Constructor & Destructor Documentation	200
12.101.2.1 Publish()	200
12.101.3 Member Function Documentation	200
12.101.3.1 getDescription()	200
12.101.3.2 getPassword()	200
12.101.3.3 getPrivacy()	200
12.101.3.4 getPublishId()	200
12.101.3.5 getSecurity()	200
12.101.3.6 getTitle()	200
12.101.3.7 operator bool()	200
12.101.3.8 setDescription()	201
12.101.3.9 setPassword()	201
12.101.3.10 setPrivacy()	201
12.101.3.11 setSecurity()	201
12.101.3.12 setTitle()	201
12.102 cbe::delegate::PublishDelegate Class Reference	202
12.102.1 Detailed Description	202
12.102.2 Member Function Documentation	202
12.102.2.1 onPublishError()	202
12.102.2.2 onPublishSuccess()	202
12.103 cbe::PublishManager Class Reference	203
12.103.1 Detailed Description	203
12.103.2 Constructor & Destructor Documentation	203
12.103.2.1 PublishManager()	203
12.103.3 Member Function Documentation	203
12.103.3.1 getPublished()	203
12.103.3.2 operator bool()	204
12.104 cbe::delegate::PublishSuccess Class Reference	204
12.104.1 Detailed Description	204
12.104.2 Member Function Documentation	204
12.104.2.1 operator bool()	204
12.105 cbe::QueryChain Class Reference	204
12.105.1 Detailed Description	205

12.105.2 Member Function Documentation	205
12.105.2.1 getResult()	206
12.105.2.2 join() [1/4]	206
12.105.2.3 join() [2/4]	206
12.105.2.4 join() [3/4]	206
12.105.2.5 join() [4/4]	207
12.106 cbe::QueryChainExt Class Reference	207
12.106.1 Detailed Description	208
12.106.2 Member Function Documentation	208
12.106.2.1 join() [1/8]	209
12.106.2.2 join() [2/8]	209
12.106.2.3 join() [3/8]	209
12.106.2.4 join() [4/8]	209
12.106.2.5 join() [5/8]	210
12.106.2.6 join() [6/8]	210
12.106.2.7 join() [7/8]	210
12.106.2.8 join() [8/8]	210
12.107 cbe::delegate::QueryDelegate Class Reference	211
12.107.1 Detailed Description	211
12.107.2 Member Function Documentation	212
12.107.2.1 onQueryError()	212
12.107.2.2 onQuerySuccess()	212
12.108 cbe::delegate::QueryError Class Reference	212
12.108.1 Detailed Description	213
12.109 cbe::delegate::QueryJoinDelegate Class Reference	213
12.109.1 Detailed Description	214
12.109.2 Member Typedef Documentation	214
12.109.2.1 ErrorInfo	214
12.109.3 Member Function Documentation	214
12.109.3.1 onQueryJoinError()	214
12.109.3.2 onQueryJoinSuccess()	214
12.110 cbe::delegate::QueryLoaded Struct Reference	215
12.110.1 Member Function Documentation	215
12.110.1.1 onQuerySuccess()	215
12.111 cbe::QueryResult Class Reference	215
12.111.1 Detailed Description	216
12.111.2 Constructor & Destructor Documentation	216
12.111.2.1 QueryResult()	216
12.111.3 Member Function Documentation	216
12.111.3.1 containersLoaded()	216
12.111.3.2 containsItem()	216
12.111.3.3 filter()	216

12.111.3.4 getItemsSnapshot()	216
12.111.3.5 itemsLoaded()	217
12.111.3.6 objectsLoaded()	217
12.111.3.7 operator bool()	217
12.111.3.8 totalCount()	217
12.112 cbe::delegate::container::RemoveDelegate Class Reference	217
12.112.1 Detailed Description	218
12.112.2 Member Function Documentation	218
12.112.2.1 onRemoveError()	218
12.112.2.2 onRemoveSuccess()	218
12.113 cbe::delegate::group::RemoveDelegate Class Reference	218
12.113.1 Detailed Description	218
12.113.2 Member Function Documentation	219
12.113.2.1 onRemoveError()	219
12.113.2.2 onRemoveSuccess()	219
12.114 cbe::delegate::object::RemoveDelegate Class Reference	219
12.114.1 Detailed Description	219
12.114.2 Member Function Documentation	219
12.114.2.1 onRemoveError()	220
12.114.2.2 onRemoveSuccess()	220
12.115 cbe::delegate::RemoveRoleDelegate Class Reference	220
12.115.1 Detailed Description	220
12.115.2 Member Function Documentation	220
12.115.2.1 onRemoveRoleError()	220
12.115.2.2 onRemoveRoleSuccess()	221
12.116 cbe::delegate::RemoveRoleMemberDelegate Class Reference	221
12.116.1 Detailed Description	221
12.116.2 Member Function Documentation	221
12.116.2.1 onRemoveRoleMemberError()	221
12.116.2.2 onRemoveRoleMemberSuccess()	221
12.117 cbe::delegate::container::RemoveSuccess Class Reference	222
12.117.1 Detailed Description	222
12.118 cbe::delegate::group::RemoveSuccess Class Reference	222
12.118.1 Detailed Description	222
12.119 cbe::delegate::object::RemoveSuccess Class Reference	222
12.119.1 Detailed Description	223
12.119.2 Member Function Documentation	223
12.119.2.1 operator bool()	223
12.120 cbe::delegate::container::RenameDelegate Class Reference	223
12.120.1 Detailed Description	223
12.120.2 Member Function Documentation	223
12.120.2.1 onRenameError()	223

12.120.2.2 onRenameSuccess()	224
12.121 cbe::delegate::group::RenameDelegate Class Reference	224
12.121.1 Detailed Description	224
12.121.2 Member Function Documentation	224
12.121.2.1 onRenameError()	224
12.121.2.2 onRenameSuccess()	224
12.122 cbe::delegate::object::RenameDelegate Class Reference	225
12.122.1 Detailed Description	225
12.122.2 Member Function Documentation	225
12.122.2.1 onRenameError()	225
12.122.2.2 onRenameSuccess()	225
12.123 cbe::delegate::group::RenameSuccess Class Reference	225
12.123.1 Detailed Description	226
12.123.2 Member Function Documentation	226
12.123.2.1 operator bool()	226
12.124 cbe::Request Struct Reference	226
12.124.1 Detailed Description	227
12.125 cbe::Role Class Reference	227
12.125.1 Detailed Description	227
12.125.2 Member Typedef Documentation	227
12.125.2.1 AddRoleMemberDelegatePtr	228
12.125.2.2 ListMembersDelegatePtr	228
12.125.2.3 RemoveRoleMemberDelegatePtr	228
12.125.3 Constructor & Destructor Documentation	228
12.125.3.1 Role()	228
12.125.4 Member Function Documentation	228
12.125.4.1 addRoleMember()	228
12.125.4.2 groupId()	228
12.125.4.3 id()	229
12.125.4.4 listMembers()	229
12.125.4.5 name()	229
12.125.4.6 operator bool()	229
12.125.4.7 removeRoleMember()	229
12.126 cbe::delegate::SearchGroupsDelegate Class Reference	229
12.126.1 Detailed Description	230
12.126.2 Member Function Documentation	230
12.126.2.1 onSearchGroupsError()	230
12.126.2.2 onSearchGroupsSuccess()	230
12.127 cbe::ShareData Struct Reference	230
12.127.1 Detailed Description	230
12.128 cbe::delegate::ShareDelegate Class Reference	231
12.128.1 Detailed Description	231

12.128.2 Member Function Documentation	231
12.128.2.1 onShareError()	231
12.128.2.2 onShareSuccess()	231
12.129 cbe::ShareManager Class Reference	232
12.129.1 Detailed Description	232
12.129.2 Member Typedef Documentation	232
12.129.2.1 ListSharesDelegatePtr	232
12.129.3 Constructor & Destructor Documentation	232
12.129.3.1 ShareManager()	232
12.129.4 Member Function Documentation	232
12.129.4.1 listAvailableShares()	233
12.129.4.2 listMyShares()	233
12.129.4.3 operator bool()	233
12.130 cbe::delegate::ShareSuccess Class Reference	233
12.130.1 Detailed Description	234
12.130.2 Member Function Documentation	234
12.130.2.1 operator bool()	234
12.131 cbe::Stream Class Reference	234
12.131.1 Detailed Description	234
12.131.2 Member Data Documentation	234
12.131.2.1 length	234
12.131.2.2 streamId	234
12.132 cbe::Subscribe Class Reference	234
12.132.1 Detailed Description	235
12.132.2 Member Function Documentation	235
12.132.2.1 getDate()	235
12.132.2.2 getDescription()	235
12.132.2.3 getOwner()	235
12.132.2.4 getPassword()	235
12.132.2.5 getPrivacy()	235
12.132.2.6 getPublishId()	236
12.132.2.7 getSecurity()	236
12.132.2.8 getSubscribeId()	236
12.132.2.9 getTitle()	236
12.132.2.10 unsubscribe()	236
12.133 cbe::delegate::SubscribeDelegate Class Reference	236
12.133.1 Detailed Description	236
12.133.2 Member Function Documentation	236
12.133.2.1 onSubscribeError()	237
12.133.2.2 onSubscribeSuccess()	237
12.134 cbe::SubscribeManager Class Reference	237
12.134.1 Detailed Description	237

12.134.2 Member Typedef Documentation	237
12.134.2.1 GetSubscriptionsDelegatePtr	238
12.134.2.2 SubscribeDelegatePtr	238
12.134.3 Member Function Documentation	238
12.134.3.1 getSubscriptions()	238
12.134.3.2 subscribe() [1/3]	238
12.134.3.3 subscribe() [2/3]	238
12.134.3.4 subscribe() [3/3]	239
12.135 cbe::delegate::TransferError Class Reference	239
12.135.1 Detailed Description	240
12.136 cbe::delegate::UnBanDelegate Class Reference	240
12.136.1 Detailed Description	241
12.136.2 Member Function Documentation	241
12.136.2.1 onUnBanError()	241
12.136.2.2 onUnBanSuccess()	241
12.137 cbe::delegate::UnBanSuccess Class Reference	241
12.137.1 Detailed Description	242
12.138 cbe::delegate::UnPublishDelegate Class Reference	242
12.138.1 Detailed Description	242
12.138.2 Member Function Documentation	242
12.138.2.1 onUnPublishError()	242
12.138.2.2 onUnPublishSuccess()	242
12.139 cbe::delegate::UnPublishSuccess Class Reference	243
12.139.1 Detailed Description	243
12.139.2 Member Function Documentation	243
12.139.2.1 operator bool()	243
12.140 cbe::delegate::UnShareDelegate Class Reference	243
12.140.1 Detailed Description	244
12.140.2 Member Function Documentation	244
12.140.2.1 onUnShareError()	244
12.140.2.2 onUnShareSuccess()	244
12.141 cbe::delegate::UnShareSuccess Class Reference	244
12.141.1 Detailed Description	244
12.141.2 Member Function Documentation	244
12.141.2.1 operator bool()	245
12.142 cbe::delegate::UnSubscribeDelegate Class Reference	245
12.142.1 Detailed Description	245
12.142.2 Member Function Documentation	245
12.142.2.1 onUnSubscribeError()	245
12.142.2.2 onUnSubscribeSuccess()	245
12.143 cbe::delegate::UnSubscribeSuccess Class Reference	246
12.143.1 Detailed Description	246

12.143.2 Member Function Documentation	246
12.143.2.1 operator bool()	246
12.144 cbe::delegate::UpdateKeyValuesDelegate Class Reference	246
12.144.1 Detailed Description	247
12.144.2 Member Function Documentation	247
12.144.2.1 onUpdateKeyValuesError()	247
12.144.2.2 onUpdateKeyValuesSuccess()	247
12.145 cbe::delegate::UploadDelegate Class Reference	247
12.145.1 Detailed Description	247
12.145.2 Member Function Documentation	248
12.145.2.1 onChunkSent()	248
12.145.2.2 onUploadError()	248
12.145.2.3 onUploadSuccess()	248
Index	249

Chapter 1

API for C++

Software Development Kit with Asynchronous API for the CloudBackend Singularity database service managing Data in Motion & Data at Rest.



For more information, see also the [CloudBackend web site](#)

Document	
Version	2.1.4
Extension	Async API
Date	2025-02-13

Copyright © CloudBackend AB

Chapter 2

Class diagrams

2.1 Overview

These are the main classes available when writing programs using the CloudBackend SDK.

The CloudBackend class is the core object and entry point into using the SDK.

The classes are represented as UML diagrams below.

2.2 CloudBackend - core object

Creating an instance of CloudBackend will be the first thing to do.

In order to initiate the core CloudBackend object, you need to call the `login()` method on the CloudBackend class. This will return the CloudBackend object instance that holds the currently logged in identity (user) and the databases available for that identity.

Once signed in the CloudBackend object will be able to access the identity's account, containers, objects and other functionality.

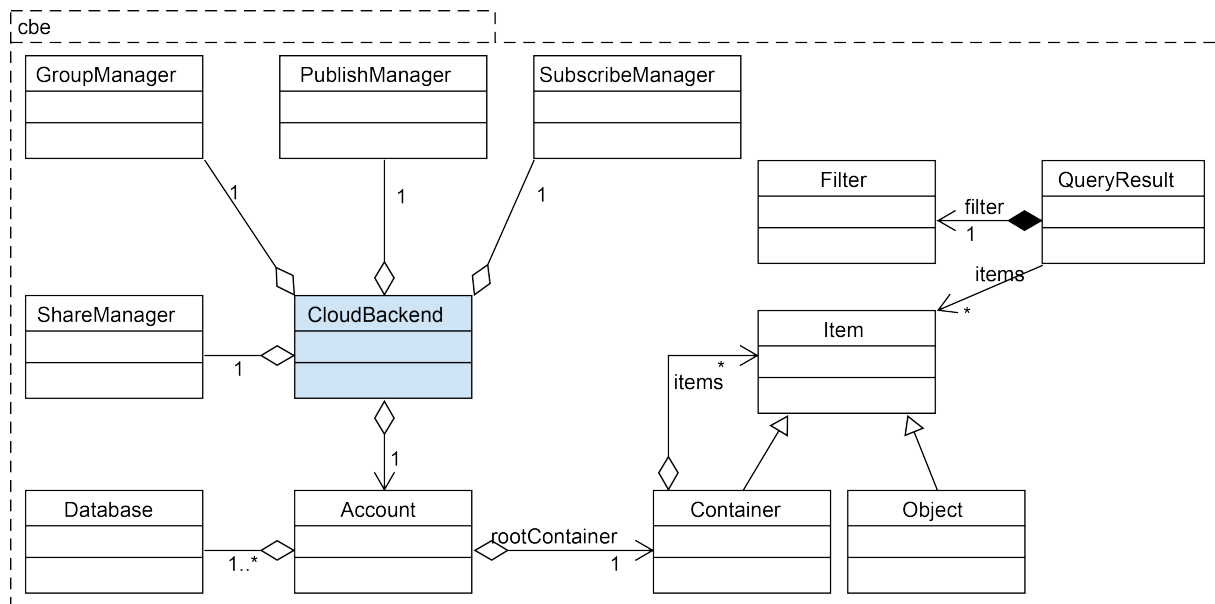


Figure 2.1 UML representation of the major classes in the SDK

2.3 Object

More details around the Object class.

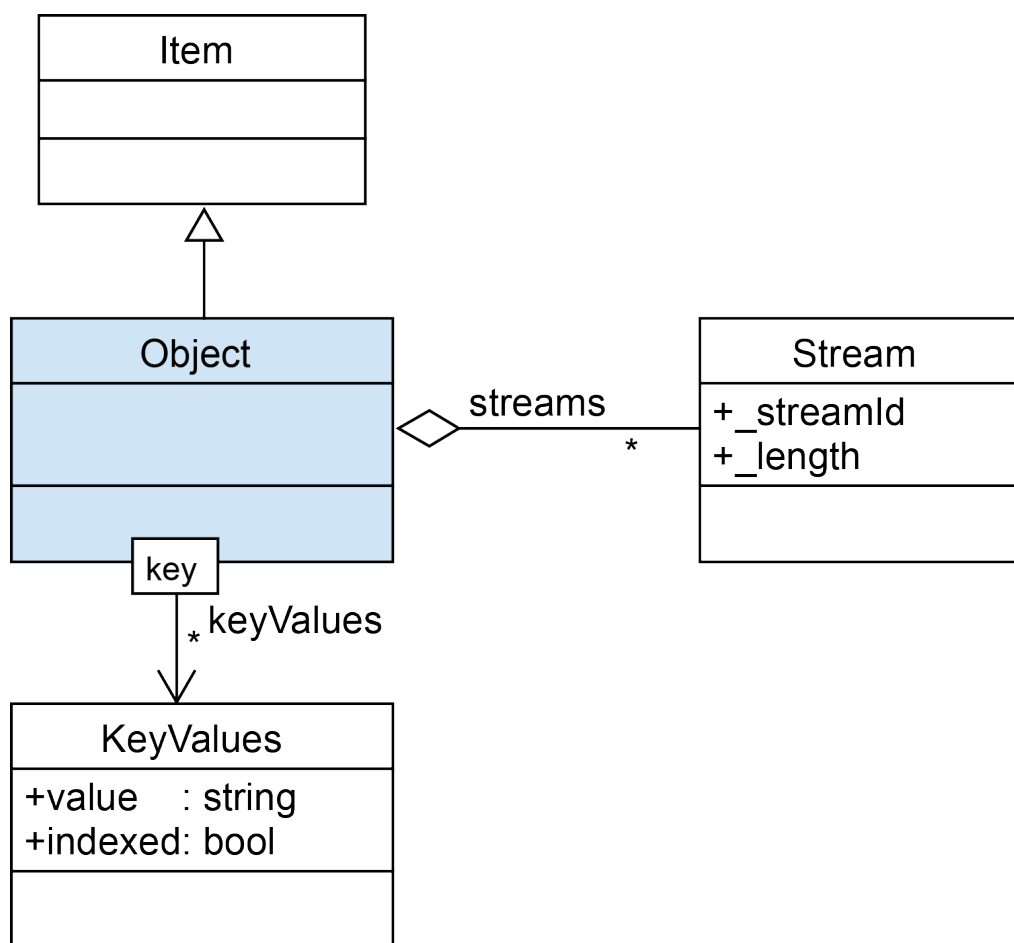


Figure 2.2 Object related classes

2.4 Group

More details around the Group class.

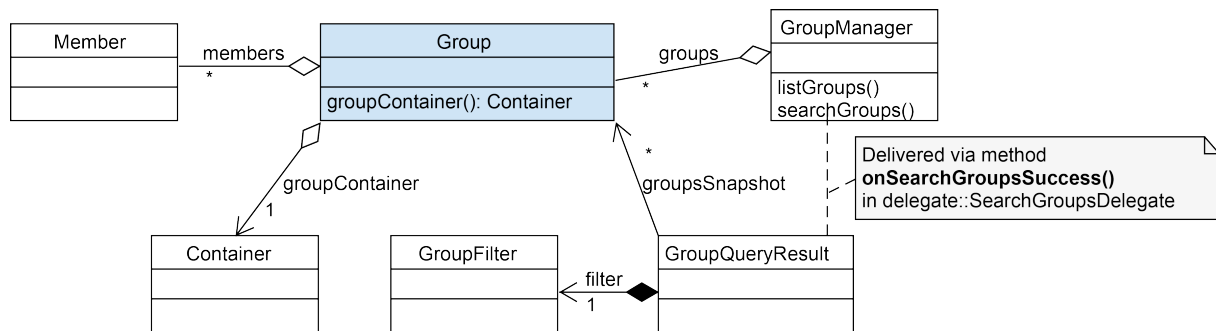


Figure 2.3 Group related classes

Chapter 3

Sync vs. Async

3.1 Choosing a Synchronous or Asynchronous coding style

Many calls to the SDK can return either a synchronous or an asynchronous response. This leads to a decision on how you wish to work based on priorities. All remote communication within the Singularity Database internally takes place asynchronously. Any updates are first made to the local cache of data, then asynchronously applied to the rest of the Singularity Database outside the SDK. When the remote request completes, a callback is made to the delegate you have implemented.

Synchronous

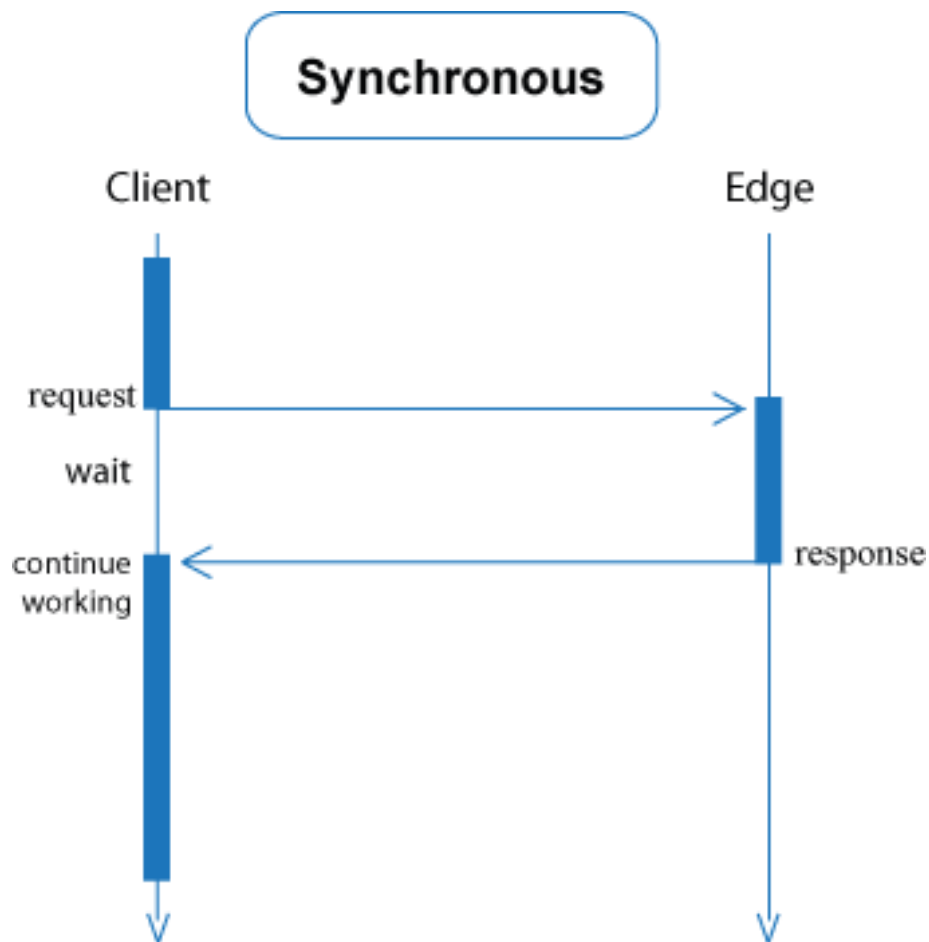


Figure 3.1 Synchronous program sequence

Working synchronously allows one to immediately act on data that has been updated. This, however, means that it is possible someone else is looking at the same data without the updates you just made (inconsistency). The Singularity Database is built on the concept of eventual consistency, meaning that all SDKs viewing or holding the same object will eventually be consistent. Additionally, if a request fails, requests dependent on the success of that request, that appear complete, will be rolled back.

Asynchronous

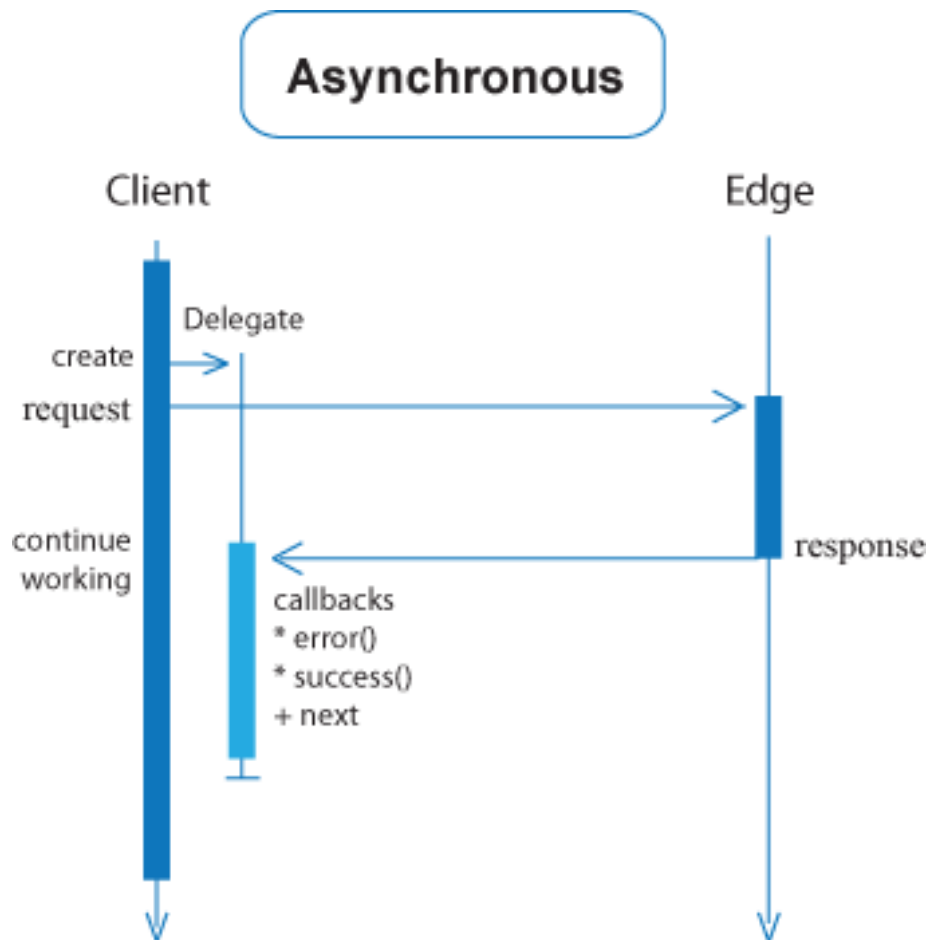


Figure 3.2 Asynchronous program sequence

Working asynchronously means that the main thread may continue to work while there is a separate parallel thread delegate created that will wait for the response, process the response, and optionally make any follow up processing or hand over. This means that the applications you write on top of the SDK will be event-driven instead of being triggered by a main loop. The end user experience will be a user interface that is responsive and does not freeze while waiting for a server response. Working this way guarantees everything presented to the end user has been propagated to the Singularity Database and will be what is retrieved if another client communicates with the database.

Pros and cons

With this SDK the developer is provided with sets of API with both Synchronous and Asynchronous variants of the calls that may communicate with the service in the cloud.

The Synchronous calls will be easier to use, but will freeze operations waiting for the cloud to respond. For a sequential program like a report or a sensor feeding data this is probably a good choice.

For a good interactive user experience in a user interface where multiple events happens in an unpredictable order, it is probably more desirable to write asynchronously on top of the SDK.

It is possible to mix Synchronous and Asynchronous calls in the same program.

Error handling

API calls have two possible outcomes: either success or error. The Synchronous API is provided in two variants that either uses exceptions or returns a result that can be of two types: either the error information or the expected result.

In the Asynchronous API the callback delegate needs to cater for both cases: success and error.

Chapter 4

Sequence diagrams - async

4.1 Login asynchronous UML

Sequence diagram for login using a delegate for asynchronous callbacks.

User can log in by calling login() submitting parameters with the credentials of:

- username
- password
- tenant

Create a Delegate that will get an input-callback of either:

- success
- error

Call the login() with credentials and the pointer to the delegate that is going to handle the callback.

Normally you need to wait for the completion and then, depending on if the result was success or error, take an appropriate action.

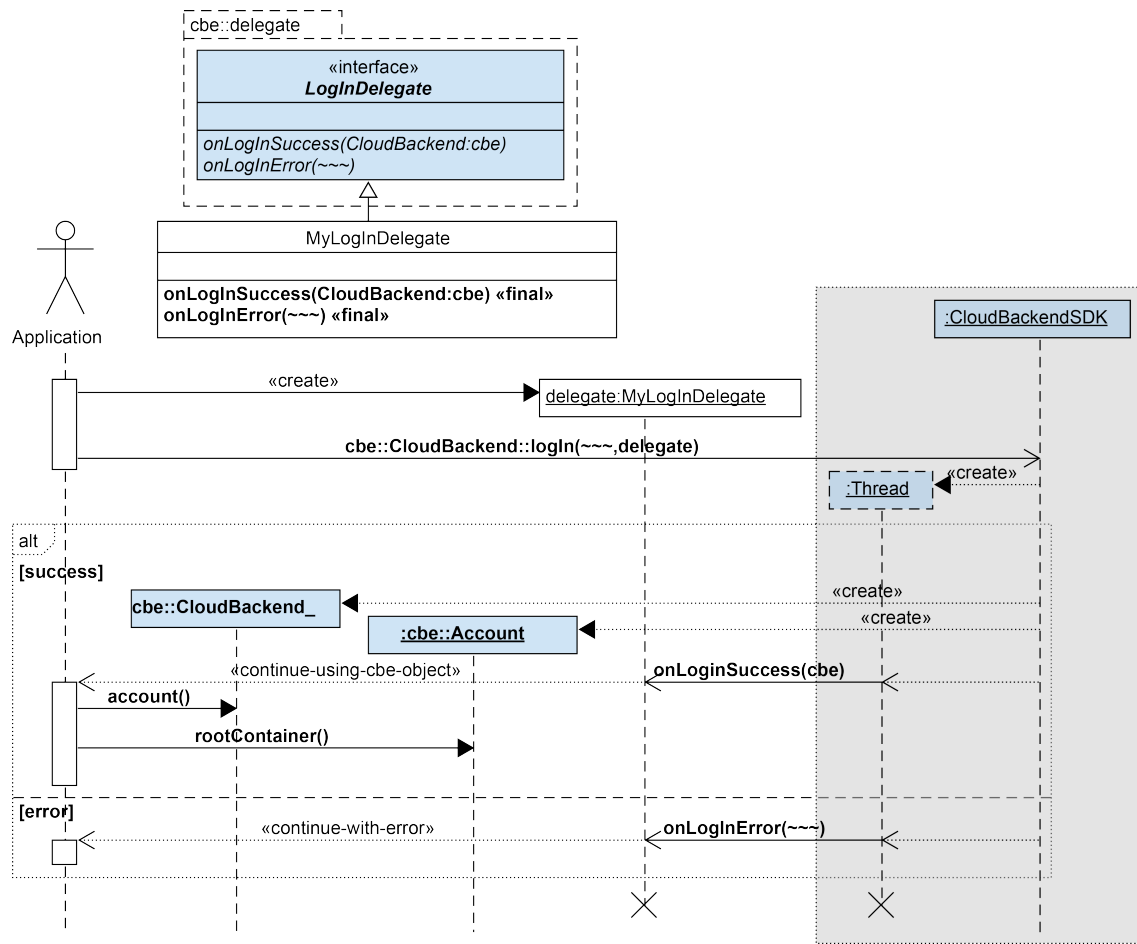


Figure 4.1 Login process sequence diagram

4.2 Query asynchronous UML

Sequence diagram for a query [optional filter] using a delegate for asynchronous callbacks.

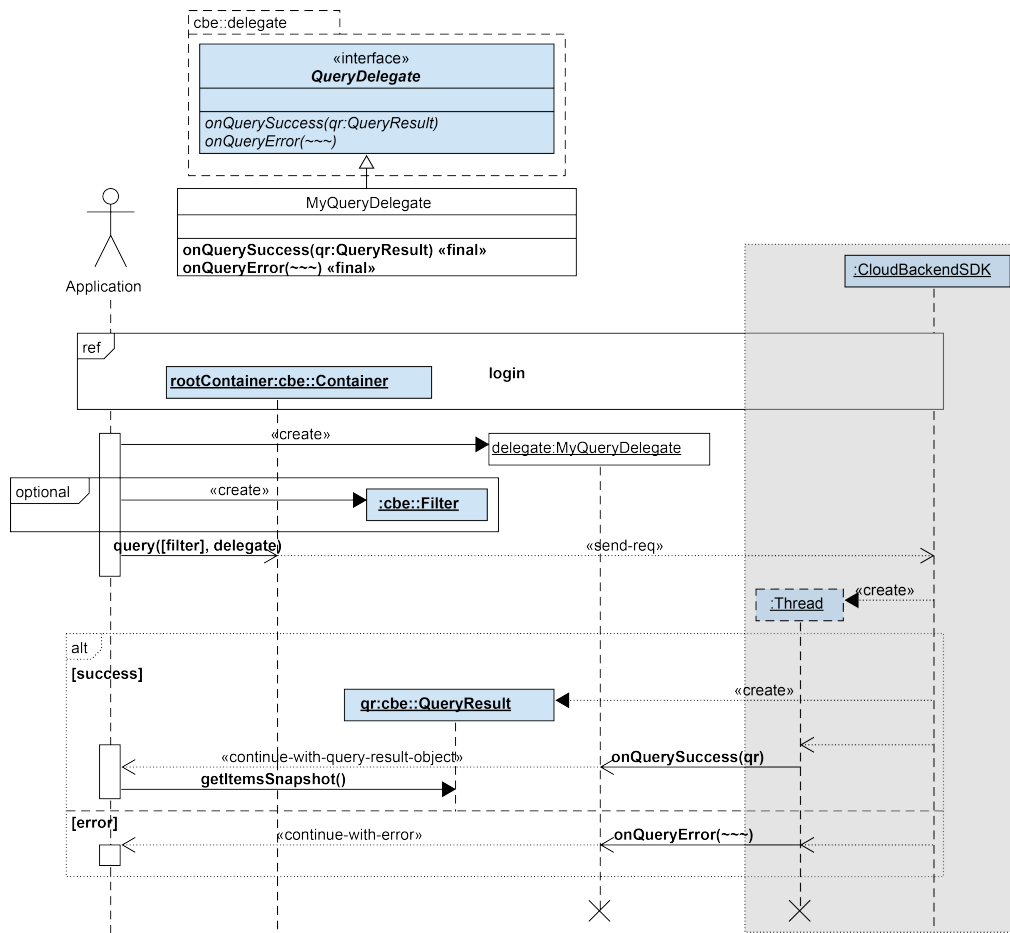


Figure 4.2 Query process sequence diagram

Chapter 5

Example code - combined lists grouped by call

Index of links — to sections that contains examples — for inspiration A line in a code example with "~~~" indicates that additional code should be added there to handle next action.

5.1 login

1. [login - async example](#)

5.2 upload

1. [upload - async example](#)

5.3 getStreams

1. [getStreams - async example](#)

5.4 uploadStream

1. [uploadStream - async example](#)

5.5 downloadStream

1. [downloadStream - async example](#)

5.6 query - select

1. [query - async example](#)
2. [QueryDelegate implementation class - async only](#)

5.7 databases - tenant

1. [databases - tenant container example](#)

5.8 Complete program

1. [all code - async example](#)

Chapter 6

Full example - async

Example program - asynchronous

```
#include <algorithm>
#include <fstream>
#include <iostream>
#include "cbe/Account.h"
#include "cbe/CloudBackend.h"
#include "cbe/Container.h"
#include "cbe/Stream.h"
#include "cbe/Object.h"
#include "cbe/QueryChain.h"
#include "cbe/util/ErrorInfo.h"
#include "MyDelegates.cpp"
#include "../user_credentials.cpp"

int main(int argc, const char** argv) {
    static constexpr const char testContainer[] = "z-MediumExample";
    cbe::Container rootContainer{cbe::DefaultCtor{}};
    cbe::Container myContainer{cbe::DefaultCtor{}};
    cbe::Filter objectFilter;
    objectFilter.setDataType(cbe::ItemType::Object);
    // ----- login_async -----
    std::shared_ptr<MyLogInDelegate> myLogInDelegate =
        std::make_shared<MyLogInDelegate>();
    cbe::CloudBackend cloudBackend = cbe::CloudBackend::logIn(username,
                                                                password,
                                                                tenant,
                                                                client,
                                                                myLogInDelegate);

    myLogInDelegate->waitForRsp();
    if (myLogInDelegate->error) {
        std::cout << "Failed to login: error={" << myLogInDelegate->errorInfo << '}'
                  << std::endl;
        cloudBackend.terminate();
        return 1;
    }
    cloudBackend = myLogInDelegate->cloudBackend;
    myLogInDelegate.reset();
    rootContainer = cloudBackend.account().rootContainer();
    std::cout << "login: " << cloudBackend.account().username() << std::endl;
    // ----- check for existing container -----
    std::shared_ptr<MyQueryDelegate> myQueryDelegate =
        std::make_shared<MyQueryDelegate>();
    rootContainer.query(myQueryDelegate).getQueryResult();
    myQueryDelegate->waitForRsp();
    cbe::QueryResult resultSet = myQueryDelegate->queryResult;
    for (auto entry : resultSet.getItemsSnapshot()) {
        if (entry.type() == cbe::ItemType::Container &&
            entry.name() == testContainer) {
            std::cout << "Using existing container:" << std::endl;
            myContainer = cbe::CloudBackend::castContainer(entry);
        }
    }
    if (!myContainer) {
        std::cout << "Creating a new container:" << std::endl;
        std::shared_ptr<MyCreateContainerDelegate> myCreateContainerDelegate =
            std::make_shared<MyCreateContainerDelegate>();
        rootContainer.createContainer(testContainer, myCreateContainerDelegate);
        myCreateContainerDelegate->waitForRsp();
        myContainer = myCreateContainerDelegate->container;
    }
    std::cout << "home://"
              << myContainer.name()
              << "/"
              << std::endl;
    // ----- upload_async -----
    constexpr const char* const uploadPath = "/tmp/";
```

```

constexpr const char* const myObjectFileName = "Medium_A_a_message";
const std::string      qualFileName = std::string(uploadPath) +
                                   myObjectFileName;

cbe::Object anObject{cbe::DefaultCtor{}};
cbe::Object myObject{cbe::DefaultCtor{}};
cbe::Object::Streams streams;
std::cout << "Create local file " << qualFileName << std::endl;
std::ofstream ofs{qualFileName};
ofs << "Line a\n"
    << "Line b\n"
    << "Line c\n"
    << std::flush;
ofs.close();
std::cout << "Uploading object from "
    << qualFileName
    << std::endl;
std::shared_ptr<MyUploadDelegate> myUploadDelegate =
    std::make_shared<MyUploadDelegate>();
myContainer.upload(qualFileName, myUploadDelegate);
myObject = myUploadDelegate->waitForRsp();
if (!myObject) {
    std::cout << "Error! " << myUploadDelegate->errorInfo << std::endl;
}
std::shared_ptr<MyGetStreamsDelegate> getStreamsDelegate =
    std::make_shared<MyGetStreamsDelegate>();
myObject.getStreams(getStreamsDelegate);
getStreamsDelegate->waitForRsp();
if (getStreamsDelegate->errorInfo) {
    std::cout << "Error! " << myLogInDelegate->errorInfo << std::endl;
}
streams = *getStreamsDelegate->streams;
// ----- uploadStream_async -----
constexpr const char* const myFile2Name = "Medium_A_an_attachment";
const std::string qualFile2Name = std::string(uploadPath) + myFile2Name;
std::ofstream ofs2{qualFile2Name};
ofs2 << "line 21\n"
    << "line 22\n"
    << "line 23\n"
    << "line 24\n"
    << std::flush;
ofs2.close();
std::cout << "Uploading additional stream from: "
    << qualFile2Name
    << std::endl;
const auto nextStreamId =
    std::max_element(std::begin(streams), std::end(streams),
        [](const cbe::Stream& stream1,
           const cbe::Stream& stream2) {
            return stream1.streamId < stream2.streamId;
        })->streamId + 1;
std::shared_ptr<MyUploadStreamDelegate> myUploadStreamDelegate =
    std::make_shared<MyUploadStreamDelegate>();
myObject.uploadStream(qualFile2Name, nextStreamId, myUploadStreamDelegate);
myUploadStreamDelegate->waitForRsp();
if (myUploadStreamDelegate->errorInfo) {
    std::cout << "Error! " << myLogInDelegate->errorInfo << std::endl;
}
std::cout << "home://"
    << myContainer.name()
    << "/"
    << myObject.name()
    << " now has two streams."
    << std::endl;
myObject.getStreams(getStreamsDelegate);
getStreamsDelegate->waitForRsp();
if (getStreamsDelegate->errorInfo) {
    std::cout << "Error! " << myLogInDelegate->errorInfo << std::endl;
}
streams = *getStreamsDelegate->streams;
// ----- downloadStream_async -----
std::cout << "Downloading stream:" << std::endl;
std::shared_ptr<MyDownloadDelegate> myDownloadDelegate =
    std::make_shared<MyDownloadDelegate>();
constexpr const char* const downloadPath = "/tmp/Medium_A_download_stream_";
for (const auto& stream : streams) {
    const auto path = std::string(downloadPath) +
        std::to_string(stream.streamId) + "_of_";
    std::cout << stream.streamId
        << " to: "
        << path
        << myObject.name()
        << std::endl;
    myObject.downloadStream(path, stream, myDownloadDelegate);
    myDownloadDelegate->waitForRsp();
    if (!myDownloadDelegate) {
        std::cout << "Error! " << myDownloadDelegate->errorInfo << std::endl;
    }
}

```

```

}
// ----- cloudbackend_query_async -----
std::cout << "Content of "
          << myContainer.name()
          << std::endl;
myContainer.query(objectFilter, myQueryDelegate);
myQueryDelegate->waitForRsp();
if (myQueryDelegate->errorInfo) {
    std::cout << "Error! " << myQueryDelegate->errorInfo << std::endl;
} else {
    cbe::QueryResult::ItemsSnapshot itemsSnapshot =
        myQueryDelegate->queryResult.getItemsSnapshot();
    std::cout << "----- table content -----" << std::endl;
    for (auto& item : itemsSnapshot) {
        anObject = cloudBackend.castObject(item);
        std::cout << " sum "
                  << anObject.length()
                  << " Bytes\t"
                  << anObject.name()
                  << std::endl;
    }
    std::cout << "-----" << std::endl;
}
std::cout << "Example Medium done" << std::endl;
cloudBackend.terminate();
} // int main()

```


Chapter 7

Document list

C++ latest version

- [HTML version](#) of the Synchronous and Asynchronous reference document
- [PDF version](#) of this reference document
- [HTML version](#) of the Asynchronous only reference document
- [PDF version](#) of the Asynchronous only reference document

Java latest version

- [Java and Android API](#) reference document

Swift latest version

- [iOS Swift API](#) reference document

Other

- [CloudBackend web site](#) document index

Chapter 8

Namespace Index

8.1 Namespace List

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

cbe	Root namespace for the CloudBackend SDK API	33
cbe::delegate	Root namespace for the delegate interfaces	40
cbe::delegate::container	Namespace for cbe::Container specific delegate interfaces	47
cbe::delegate::group	Namespace for cbe::Group specific delegate interfaces	48
cbe::delegate::object	Namespace for cbe::Object specific delegate interfaces	49
cbe::util	General support program utilities	50

Chapter 9

Hierarchical Index

9.1 Class Hierarchy

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

cbe::Account	51
cbe::delegate::AclDelegate	53
cbe::delegate::AddRoleMemberDelegate	54
cbe::delegate::BanDelegate	56
cbe::delegate::BanSuccess	57
cbe::delegate::ChunkTransferred	57
cbe::CloudBackend	58
cbe::util::Context	83
cbe::delegate::CreateAccountDelegate	83
cbe::delegate::CreateContainerDelegate	84
cbe::delegate::CreateGroupDelegate	85
cbe::delegate::CreateObjectDelegate	86
cbe::delegate::CreateRoleDelegate	87
cbe::Database	88
cbe::delegate::DownloadBinaryDelegate	90
cbe::delegate::DownloadBinarySuccess	91
cbe::delegate::DownloadDelegate	92
cbe::delegate::DownloadSuccess	93
cbe::delegate::Error	95
cbe::delegate::QueryError	212
cbe::delegate::TransferError	239
cbe::delegate::ItemError::Error	96
cbe::delegate::JoinError::Error	97
cbe::delegate::LoadError::Error	97
cbe::util::ErrorInfo	140
cbe::util::ErrorInfoImpl< Error >	141
cbe::delegate::AclDelegate::ErrorInfo	98
cbe::delegate::AddRoleMemberDelegate::ErrorInfo	99
cbe::delegate::BanDelegate::ErrorInfo	100
cbe::delegate::CreateAccountDelegate::ErrorInfo	105
cbe::delegate::CreateContainerDelegate::ErrorInfo	106
cbe::delegate::CreateGroupDelegate::ErrorInfo	107
cbe::delegate::CreateObjectDelegate::ErrorInfo	108
cbe::delegate::CreateRoleDelegate::ErrorInfo	109
cbe::delegate::DownloadBinaryDelegate::ErrorInfo	110
cbe::delegate::DownloadDelegate::ErrorInfo	111
cbe::delegate::GetStreamsDelegate::ErrorInfo	112
cbe::delegate::GetSubscriptionsDelegate::ErrorInfo	113
cbe::delegate::KickDelegate::ErrorInfo	117
cbe::delegate::LeaveDelegate::ErrorInfo	118

cbe::delegate::ListGroupDelegate::ErrorInfo	119
cbe::delegate::ListMembersDelegate::ErrorInfo	120
cbe::delegate::ListRolesDelegate::ErrorInfo	121
cbe::delegate::ListSharesDelegate::ErrorInfo	122
cbe::delegate::LoginDelegate::ErrorInfo	123
cbe::delegate::PublishDelegate::ErrorInfo	127
cbe::delegate::QueryDelegate::ErrorInfo	128
cbe::delegate::RemoveRoleDelegate::ErrorInfo	129
cbe::delegate::RemoveRoleMemberDelegate::ErrorInfo	130
cbe::delegate::SearchGroupsDelegate::ErrorInfo	131
cbe::delegate::ShareDelegate::ErrorInfo	132
cbe::delegate::SubscribeDelegate::ErrorInfo	133
cbe::delegate::UnBanDelegate::ErrorInfo	134
cbe::delegate::UnPublishDelegate::ErrorInfo	135
cbe::delegate::UnShareDelegate::ErrorInfo	136
cbe::delegate::UnSubscribeDelegate::ErrorInfo	137
cbe::delegate::UpdateKeyValuesDelegate::ErrorInfo	138
cbe::delegate::UploadDelegate::ErrorInfo	139
cbe::delegate::container::MoveDelegate::ErrorInfo	102
cbe::delegate::container::RemoveDelegate::ErrorInfo	103
cbe::delegate::container::RenameDelegate::ErrorInfo	104
cbe::delegate::group::JoinDelegate::ErrorInfo	114
cbe::delegate::group::RemoveDelegate::ErrorInfo	115
cbe::delegate::group::RenameDelegate::ErrorInfo	116
cbe::delegate::object::MoveDelegate::ErrorInfo	124
cbe::delegate::object::RemoveDelegate::ErrorInfo	125
cbe::delegate::object::RenameDelegate::ErrorInfo	126
cbe::util::ErrorInfoImpl< ErrorT >	141
cbe::Filter	145
cbe::delegate::GetStreamsDelegate	150
cbe::delegate::GetSubscriptionsDelegate	151
cbe::Group	151
cbe::GroupFilter	158
cbe::GroupManager	160
cbe::GroupQueryResult	162
cbe::delegate::ICloudBackendListener	164
cbe::delegate::CloudBackendListenerDelegate	69
cbe::Item	166
cbe::Container	72
cbe::Object	186
cbe::delegate::ItemError	170
cbe::delegate::group::JoinDelegate	171
cbe::delegate::JoinDelegate	171
cbe::delegate::JoinError	172
cbe::delegate::KickDelegate	173
cbe::delegate::KickSuccess	174
cbe::delegate::LeaveDelegate	174
cbe::delegate::LeaveSuccess	175
cbe::delegate::ListGroupDelegate	176
cbe::delegate::ListMembersDelegate	177
cbe::delegate::ListRolesDelegate	177
cbe::delegate::ListSharesDelegate	178
cbe::delegate::LoadError	179
cbe::delegate::LoginDelegate	180
cbe::MatchesID< IdT >	181
cbe::Member	181
cbe::delegate::container::MoveDelegate	184
cbe::delegate::object::MoveDelegate	185

cbe::util::Optional< T >	197
cbe::Publish	199
cbe::delegate::PublishDelegate	202
cbe::PublishManager	203
cbe::delegate::PublishSuccess	204
cbe::QueryChain	204
cbe::QueryChainExt	207
cbe::delegate::QueryDelegate	211
cbe::delegate::QueryJoinDelegate	213
cbe::delegate::QueryLoaded	215
cbe::QueryResult	215
cbe::delegate::container::RemoveDelegate	217
cbe::delegate::group::RemoveDelegate	218
cbe::delegate::object::RemoveDelegate	219
cbe::delegate::RemoveRoleDelegate	220
cbe::delegate::RemoveRoleMemberDelegate	221
cbe::delegate::container::RemoveSuccess	222
cbe::delegate::group::RemoveSuccess	222
cbe::delegate::object::RemoveSuccess	222
cbe::delegate::container::RenameDelegate	223
cbe::delegate::group::RenameDelegate	224
cbe::delegate::object::RenameDelegate	225
cbe::delegate::group::RenameSuccess	225
cbe::Request	226
cbe::Role	227
std::runtime_error	
cbe::util::Exception	143
cbe::QueryChain::Exception	142
cbe::util::ExceptionImpl< ErrorInfoT >	144
cbe::util::Optional< T >::BadAccess	55
cbe::delegate::SearchGroupsDelegate	229
cbe::ShareData	230
cbe::delegate::ShareDelegate	231
cbe::ShareManager	232
cbe::delegate::ShareSuccess	233
cbe::Stream	234
cbe::Subscribe	234
cbe::delegate::SubscribeDelegate	236
cbe::SubscribeManager	237
cbe::delegate::UnBanDelegate	240
cbe::delegate::UnBanSuccess	241
cbe::delegate::UnPublishDelegate	242
cbe::delegate::UnPublishSuccess	243
cbe::delegate::UnShareDelegate	243
cbe::delegate::UnShareSuccess	244
cbe::delegate::UnSubscribeDelegate	245
cbe::delegate::UnSubscribeSuccess	246
cbe::delegate::UpdateKeyValuesDelegate	246
cbe::delegate::UploadDelegate	247

Chapter 10

Class Index

10.1 Class List

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

cbe::Account	
Login account information	51
cbe::delegate::AclDelegate	53
cbe::delegate::AddRoleMemberDelegate	54
cbe::util::Optional< T >::BadAccess	
Thrown by value() method when accessing an object that does not contain a value	55
cbe::delegate::BanDelegate	56
cbe::delegate::BanSuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::BanDelegate::onBanSuccess	57
cbe::delegate::ChunkTransferred	
Bundles the progress event information in connection with download and upload	57
cbe::CloudBackend	
The session that holds the connection with the cloud	58
cbe::delegate::CloudBackendListenerDelegate	69
cbe::Container	
A collection of Item , can also represent a table or folder	72
cbe::util::Context	83
cbe::delegate::CreateAccountDelegate	83
cbe::delegate::CreateContainerDelegate	84
cbe::delegate::CreateGroupDelegate	85
cbe::delegate::CreateObjectDelegate	86
cbe::delegate::CreateRoleDelegate	87
cbe::Database	
A database	88
cbe::delegate::DownloadBinaryDelegate	90
cbe::delegate::DownloadBinarySuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::DownloadBinaryDelegate::onDownloadBinarySuccess	91
cbe::delegate::DownloadDelegate	92
cbe::delegate::DownloadSuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::DownloadDelegate::onDownloadSuccess	93
cbe::delegate::Error	95
cbe::delegate::ItemError::Error	96
cbe::delegate::JoinError::Error	97
cbe::delegate::LoadError::Error	97
cbe::delegate::AclDelegate::ErrorInfo	98
cbe::delegate::AddRoleMemberDelegate::ErrorInfo	99
cbe::delegate::BanDelegate::ErrorInfo	100

cbe::delegate::container::MoveDelegate::ErrorInfo	102
cbe::delegate::container::RemoveDelegate::ErrorInfo	103
cbe::delegate::container::RenameDelegate::ErrorInfo	104
cbe::delegate::CreateAccountDelegate::ErrorInfo	105
cbe::delegate::CreateContainerDelegate::ErrorInfo	106
cbe::delegate::CreateGroupDelegate::ErrorInfo	107
cbe::delegate::CreateObjectDelegate::ErrorInfo	108
cbe::delegate::CreateRoleDelegate::ErrorInfo	109
cbe::delegate::DownloadBinaryDelegate::ErrorInfo	110
cbe::delegate::DownloadDelegate::ErrorInfo	111
cbe::delegate::GetStreamsDelegate::ErrorInfo	112
cbe::delegate::GetSubscriptionsDelegate::ErrorInfo	113
cbe::delegate::group::JoinDelegate::ErrorInfo	114
cbe::delegate::group::RemoveDelegate::ErrorInfo	115
cbe::delegate::group::RenameDelegate::ErrorInfo	116
cbe::delegate::KickDelegate::ErrorInfo	117
cbe::delegate::LeaveDelegate::ErrorInfo	118
cbe::delegate::ListGroupDelegate::ErrorInfo	119
cbe::delegate::ListMembersDelegate::ErrorInfo	120
cbe::delegate::ListRolesDelegate::ErrorInfo	121
cbe::delegate::ListSharesDelegate::ErrorInfo	122
cbe::delegate::LoginDelegate::ErrorInfo	123
cbe::delegate::object::MoveDelegate::ErrorInfo	124
cbe::delegate::object::RemoveDelegate::ErrorInfo	125
cbe::delegate::object::RenameDelegate::ErrorInfo	126
cbe::delegate::PublishDelegate::ErrorInfo	127
cbe::delegate::QueryDelegate::ErrorInfo	128
cbe::delegate::RemoveRoleDelegate::ErrorInfo	129
cbe::delegate::RemoveRoleMemberDelegate::ErrorInfo	130
cbe::delegate::SearchGroupsDelegate::ErrorInfo	131
cbe::delegate::ShareDelegate::ErrorInfo	132
cbe::delegate::SubscribeDelegate::ErrorInfo	133
cbe::delegate::UnBanDelegate::ErrorInfo	134
cbe::delegate::UnPublishDelegate::ErrorInfo	135
cbe::delegate::UnShareDelegate::ErrorInfo	136
cbe::delegate::UnSubscribeDelegate::ErrorInfo	137
cbe::delegate::UpdateKeyValuesDelegate::ErrorInfo	138
cbe::delegate::UploadDelegate::ErrorInfo	139
cbe::util::ErrorInfo	140
cbe::util::ErrorInfoImpl< ErrorT >	141
cbe::QueryChain::Exception	142
cbe::util::Exception	143
cbe::util::ExceptionImpl< ErrorInfoT >	144
cbe::Filter	
Use to select Item that meets specific criterias when doing a query	145
cbe::delegate::GetStreamsDelegate	150
cbe::delegate::GetSubscriptionsDelegate	151
cbe::Group	
A group of members	151
cbe::GroupFilter	
To filter when searching a list of Group	158
cbe::GroupManager	
For managing the groups	160
cbe::GroupQueryResult	
Resultset of data retrieved in a search for Group	162
cbe::delegate::ICloudBackendListener	164
cbe::Item	
A set made up of Container and Object	166

cbe::delegate::ItemError	170
cbe::delegate::group::JoinDelegate	171
cbe::delegate::JoinDelegate	171
cbe::delegate::JoinError	172
cbe::delegate::KickDelegate	173
cbe::delegate::KickSuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::KickDelegate::onKickSuccess	
174	
cbe::delegate::LeaveDelegate	174
cbe::delegate::LeaveSuccess	175
cbe::delegate::ListGroupDelegate	176
cbe::delegate::ListMembersDelegate	177
cbe::delegate::ListRolesDelegate	177
cbe::delegate::ListSharesDelegate	178
cbe::delegate::LoadError	179
cbe::delegate::LoginDelegate	180
cbe::MatchesID< IdT >	
Search lists with ID	181
cbe::Member	
A of member of a Group	181
cbe::delegate::container::MoveDelegate	184
cbe::delegate::object::MoveDelegate	185
cbe::Object	
Holder of a set of data, can represent a table row	186
cbe::util::Optional< T >	
Class template Optional manages an optional contained value - i.e., a value that is either present or absent	197
cbe::Publish	
Managing a published Item	199
cbe::delegate::PublishDelegate	202
cbe::PublishManager	
Managing the list of web shares	203
cbe::delegate::PublishSuccess	
Convenience type that bundles the parameter passed to method cbe::delegate::ShareDelegate::onShareSuccess	
204	
cbe::QueryChain	
To do a search for Object combining more than one Container table	204
cbe::QueryChainExt	
Extension of class QueryChain	207
cbe::delegate::QueryDelegate	211
cbe::delegate::QueryError	212
cbe::delegate::QueryJoinDelegate	213
cbe::delegate::QueryLoaded	215
cbe::QueryResult	
Resultset of data retrieved	215
cbe::delegate::container::RemoveDelegate	217
cbe::delegate::group::RemoveDelegate	218
cbe::delegate::object::RemoveDelegate	219
cbe::delegate::RemoveRoleDelegate	220
cbe::delegate::RemoveRoleMemberDelegate	221
cbe::delegate::container::RemoveSuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::container::RemoveDelegate::onRemoveSuccess	
222	
cbe::delegate::group::RemoveSuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::group::RemoveDelegate::onRemoveSuccess	
222	

cbe::delegate::object::RemoveSuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::object::onRemoveSuccess	
RemoveSuccess	222
cbe::delegate::container::RenameDelegate	223
cbe::delegate::group::RenameDelegate	224
cbe::delegate::object::RenameDelegate	225
cbe::delegate::group::RenameSuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::group::RenameDelegate::onRenameSuccess	
RenameSuccess	225
cbe::Request	226
cbe::Role	
User role information	227
cbe::delegate::SearchGroupsDelegate	229
cbe::ShareData	230
cbe::delegate::ShareDelegate	231
cbe::ShareManager	
Managing Shares	232
cbe::delegate::ShareSuccess	
Convenience type that bundles the parameter passed to method cbe::delegate::ShareDelegate::onShareSuccess	
ShareSuccess	233
cbe::Stream	
A data file attached to Object	234
cbe::Subscribe	
Managing a subscribed Item	234
cbe::delegate::SubscribeDelegate	236
cbe::SubscribeManager	
For managing subscriptions	237
cbe::delegate::TransferError	239
cbe::delegate::UnBanDelegate	240
cbe::delegate::UnBanSuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::UnBanDelegate::onUnBanSuccess	
UnBanSuccess	241
cbe::delegate::UnPublishDelegate	242
cbe::delegate::UnPublishSuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::UnPublishDelegate::onUnPublishError	
UnPublishSuccess	243
cbe::delegate::UnShareDelegate	243
cbe::delegate::UnShareSuccess	
Convenience type that bundles the parameter passed to method cbe::delegate::ShareDelegate::onShareSuccess	
UnShareSuccess	244
cbe::delegate::UnSubscribeDelegate	245
cbe::delegate::UnSubscribeSuccess	
Convenience type that bundles all parameters passed to method cbe::delegate::UnSubscribeDelegate::onUnSubscribeSuccess	
UnSubscribeSuccess	246
cbe::delegate::UpdateKeyValuesDelegate	246
cbe::delegate::UploadDelegate	247

Chapter 11

Namespace Documentation

11.1 cbe Namespace Reference

Root namespace for the CloudBackend SDK API.

Namespaces

- [delegate](#)
Root namespace for the delegate interfaces.
- [util](#)
General support program utilities.

Classes

- class [Account](#)
Login account information.
- class [CloudBackend](#)
The session that holds the connection with the cloud.
- class [Container](#)
A collection of [Item](#), can also represent a table or folder.
- class [Database](#)
A database.
- class [Filter](#)
Use to select [Item](#) that meets specific criterias when doing a query.
- class [Group](#)
A group of members.
- class [GroupFilter](#)
To filter when searching a list of [Group](#).
- class [GroupManager](#)
For managing the groups.
- class [GroupQueryResult](#)
Resultset of data retrieved in a search for [Group](#).
- class [Item](#)
A set made up of [Container](#) and [Object](#).
- struct [Request](#)
- class [Member](#)
A of member of a [Group](#).
- class [Object](#)
Holder of a set of data, can represent a table row.
- class [Publish](#)

- Managing a published [Item](#).*
- class [PublishManager](#)
 - Managing the list of web shares.*
- class [QueryChain](#)
 - To do a search for [Object](#) combining more than one [Container](#) table.*
- class [QueryChainExt](#)
 - Extension of class [QueryChain](#).*
- class [QueryResult](#)
 - resultset of data retrieved.*
- class [Role](#)
 - User role information.*
- class [ShareManager](#)
 - Managing Shares.*
- class [Stream](#)
 - A data file attached to [Object](#).*
- class [Subscribe](#)
 - Managing a subscribed [Item](#).*
- class [SubscribeManager](#)
 - For managing subscriptions.*
- struct [ShareData](#)
- class [MatchesID](#)
 - Search lists with ID.*

Typedefs

- using [Streams](#) = std::vector< [cbe::Stream](#) >
 - Collection of [Stream](#) objects.*
- using [AclGroupId](#) = std::uint64_t
 - Identifies ACL-group.*
- using [ContainerId](#) = std::uint64_t
 - Unique Id of a [cbe::Container](#).*
- using [DatabaseId](#) = std::uint64_t
 - The id of a [Database](#).*
- using [Date](#) = std::uint64_t
 - A time-stamp in the unix epoch format.*
- using [GroupId](#) = std::uint64_t
 - Uniquely identifies the [Group](#).*
- using [ItemId](#) = std::uint64_t
 - Id of a [cbe::Container](#) or [cbe::Object](#).*
- using [MemberId](#) = std::uint64_t
 - Represents the [cbe::Group](#) membership id.*
- using [ListenerHandle](#) = std::uint64_t
 - Unique Id for added Listener.*
- using [ObjectId](#) = std::uint64_t
 - Unique Id of a [cbe::Object](#).*
- using [PublishId](#) = std::uint64_t
 - Id of a subscribed [cbe::Container](#) or [cbe::Object](#).*
- using [RoleId](#) = std::uint64_t
 - Uniquely identifies the [Role](#).*
- using [ShareId](#) = std::uint64_t
 - Uniquely identifies a sharing of a [cbe::Container](#) or [cbe::Object](#).*

- using [StreamId](#) = std::uint64_t
Uniquely identifies a [cbe::Stream](#).
- using [Subscribeld](#) = std::uint64_t
Id of a subscription of a [cbe::Container](#) or [cbe::Object](#).
- using [UserId](#) = std::uint64_t
Uniquely identifies the CBE user number.
- using **account_status_t** = std::uint32_t
- using **failed_status_t** = std::uint32_t
- using **http_t** = std::uint32_t
- using **publish_access_t** = std::uint32_t
- using **publish_visibility_t** = std::uint32_t
- using **sync_direction_t** = std::uint32_t
- using **sync_status_t** = std::uint32_t
- using **transfer_t** = std::uint32_t
- using [ErrorCode](#) = std::uint32_t
Mimics the general error code encoding in the www.
- using **application_t** = int
- using **item_t** = int
- using **object_t** = int
- using **permission_status_t** = int
- using **stream_t** = int
- using **visibility** = int
- using [AclMap](#) = std::map< [cbe::AclGroupId](#), std::pair< [cbe::Permissions](#), [AclScope](#) > >
*ACL map (**Access Control List**) relating to users and groups.*
- using [DataBases](#) = std::map< std::string, [cbe::Database](#) >
Databases available for the [account](#).
- using [Items](#) = std::vector< [cbe::Item](#) >
Collection of [items](#).
- using [KeyValues](#) = std::map< std::string, std::pair< std::string, bool > >
Map with key/value pairs, a.k.a. metadata.
- using [MemberBanInfo](#) = std::map< std::pair< [cbe::MemberId](#), [cbe::MemberId](#) >, std::pair< [cbe::Date](#), std::string > >
Map of a pair of [members](#), associated with ban information.
- using [ShareIds](#) = std::map< [cbe::ShareId](#), std::vector< [cbe::ShareData](#) > >
Map of [cbe::ShareData](#) for a specific [cbe::ShareId](#).

Enumerations

- enum class [AccountStatus](#) : [cbe::account_status_t](#) { **NotLoggedIn** = 1 , **LoggedIn** = 2 , **Failed** = 3 }
- enum class [AclScope](#) : [uint64_t](#) { **User** = 1 , **Group** = 2 , **Role** = 3 }
- enum [DefaultCtor](#)
Default constructor marker.
- enum class [FilterOrder](#) : [uint32_t](#) {
 Title = 1 , **Relevance** = 2 , **Published** = 3 , **Updated** = 4 ,
 Length = 5 , **S1** = 6 , **S2** = 7 , **S3** = 8 ,
 S4 = 9 }
- enum class [ItemType](#) : [cbe::item_t](#) {
 Unapplicable = 1 , **Unknown** = 2 , **Object** = 4 , **Container** = 8 ,
 Tag = 16 , **Group** = 32 }
- enum class [ObjectType](#) : [object_t](#) { **Other** = 1 , **GroupInvites** = 2 , **ShareInvite** = 3 }

- enum class [Permissions](#) : permission_status_t {
Read = 1 , **Write** = 2 , **ReadWrite** = 3 , **Delete** = 4 ,
ReadDelete = 5 , **WriteDelete** = 6 , **ReadWriteDelete** = 7 , **ChangeACL** = 8 ,
ReadChangeACL = 9 , **WriteChangeACL** = 10 , **ReadWriteChangeACL** = 11 , **DeleteChangeACL** = 12 ,
ReadDeleteChangeACL = 13 , **WriteDeleteChangeACL** = 14 , **AllPermissions** = 15 , **NoPermissions** = 0
}

Represents the access permission that can be set for any [cbe::Object](#) or [cbe::Container](#).

- enum class [PublishAccess](#) : cbe::publish_access_t { **Read** = 1 , **Update** = 2 , **Create** = 3 }
- enum class [PublishVisibility](#) : cbe::publish_visibility_t { **Public** = 1 , **Friends** = 2 , **Private** = 3 , **Invited** = 4 }
- enum class [Visibility](#) : visibility { **Public** = 1 , **Private** = 2 }

Functions

- [cbe::ItemType](#) **operator|** ([cbe::ItemType](#) lh, [cbe::ItemType](#) rh)
- std::ostream & **operator<<** (std::ostream &os, [ItemType](#) itemType)

11.1.1 Detailed Description

Root namespace for the CloudBackend SDK API.

Located in [Types.h](#) that should be included in programs.

11.1.2 Typedef Documentation

11.1.2.1 AclGroupId

using [cbe::AclGroupId](#) = typedef std::uint64_t

Identifies ACL-group.

Representation that can be a UserId or GroupId.

I.e., the union set:

$\text{AclGroupId} = \text{UserId} \cup \text{GroupId}$.

11.1.2.2 AclMap

using [cbe::AclMap](#) = typedef std::map<[cbe::AclGroupId](#), std::pair<[cbe::Permissions](#), [AclScope](#)> >

ACL map (**Access Control List**) relating to users and groups.

Map of a specific user or group, in terms of [cbe::UserId](#) or [cbe::GroupId](#) and its [cbe::Permission](#) on a [Container](#) or [Object](#), to represent access permissions for specific users or groups.

11.1.2.3 Date

using [cbe::Date](#) = typedef std::uint64_t

A time-stamp in the unix epoch format.

A.k.a. POSIX time or time-stamp.

The equivalent in Linux shell can be found using e.g.,

```
date +%s --utc --date='now'
```

```
date +%s --utc --date='today 17:30:00'
```

To convert a time-stamp to human readable format use e.g.,

```
date --utc --date=@1678902345
```

11.1.2.4 ErrorCode

using [cbe::ErrorCode](#) = typedef std::uint32_t

Mimics the general error code encoding in the www.

see [Wikipedia: List of HTTP status codes](#)

11.1.2.5 KeyValues

```
using cbe::KeyValues = typedef std::map<std::string, std::pair<std::string, bool> >
```

Map with key/value pairs, a.k.a. metadata.

Can be applied on [cbe::Object](#) but not on [cbe::Container](#).

- **key** uniquely identifies the value; name must start with a letter or _
- **value** and **index flag**
 - a string value
 - a boolean flag indicating whether this key/value entry is index (`true`) or not indexed (`false`).

11.1.2.6 MemberBanInfo

```
using cbe::MemberBanInfo = typedef std::map<std::pair<cbe::MemberId, cbe::MemberId>, std::pair<cbe::Date, std::string> >
```

Map of a pair of [members](#), associated with ban information.

Structure:

- **Key** is formed by a `std::pair` of [members](#), where:
 - `first` represents the banned member.
 - `second` represents the banning member with administration privileges.
- **Value** is formed by another pair:
 - `date` of the ban
 - a free text reason message.

11.1.3 Enumeration Type Documentation

11.1.3.1 AccountStatus

```
enum cbe::AccountStatus : cbe::account_status_t [strong]
```

Callback for login returns one of these three in callback.

1. NotLoggedIn
2. LoggedIn
3. Failed

11.1.3.2 AclScope

```
enum cbe::AclScope : uint64_t [strong]
```

Enum that determines different ACL categories

11.1.3.3 DefaultCtor

enum `cbe::DefaultCtor`

Default constructor marker.

To default construct objects from most of the CloudBackend classes, this marker type is required.

Example use

```
~~~  
// Conceptually, a default construction of an instance of cbe::Container  
cbe::Container myContainer{ cbe::DefaultCtor{ } };  
// Ditto  
cbe::Object myObject{ cbe::DefaultCtor{ } };  
~~~
```

11.1.3.4 FilterOrder

enum `cbe::FilterOrder` : `uint32_t` [strong]

Set the filter order in which the search or query will be sorted after.

1. Title
2. Relevance
3. Published
4. Updated
5. Length
6. S1
7. S2
8. S3
9. S4

11.1.3.5 ItemType

enum `cbe::ItemType` : `cbe::item_t` [strong]

ItemType can be used to sort out cbe objects if the user would like to create a container to put all different kinds of cbe objects in. E.g.,

- `Object`
- `Container`
- `Group`

11.1.3.6 ObjectType

enum `cbe::ObjectType` : `object_t` [strong]

Type of invite.

11.1.3.7 Permissions

```
enum cbe::Permissions : permission_status_t [strong]
```

Represents the access permission that can be set for any [cbe::Object](#) or [cbe::Container](#).

Forms the the possible different combinations of:

- 1: Read
- 2: Write
- 4: Delete
- 8: ChangeACL

combinations give access for users to use different API calls.

0. NoPermissions

1. Read
2. Write
3. ReadWrite
4. Delete
5. ReadDelete
6. WriteDelete
7. ReadWriteDelete
8. ChangeACL
9. ReadChangeACL
10. WriteChangeACL
11. ReadWriteChangeACL
12. DeleteChangeACL
13. ReadDeleteChangeACL
14. WriteDeleteChangeACL
15. AllPermissions

E.g.: `cbe::Permissions::ReadDelete` gives the ability to call `move()` on a [Container](#) or an [Object](#).

11.1.3.8 PublishAccess

```
enum cbe::PublishAccess : cbe::publish_access_t [strong]
```

Access permission for publish

1. Read
2. Update
3. Create

11.1.3.9 PublishVisibility

```
enum cbe::PublishVisibility : cbe::publish_visibility_t [strong]
```

Visibility for publish

1. Public
2. Friends
3. Private
4. Invited

11.1.3.10 Visibility

```
enum cbe::Visibility : visibility [strong]
```

Visibility is used for both groups and members, in this version the member visibility will be Public for all members who join a group.

Members will in the future also have the option of visibility friends.

1. Public
2. Private

11.2 cbe::delegate Namespace Reference

Root namespace for the delegate interfaces.

Namespaces

- [container](#)
Namespace for [cbe::Container](#) specific delegate interfaces.
- [group](#)
Namespace for [cbe::Group](#) specific delegate interfaces.
- [object](#)
Namespace for [cbe::Object](#) specific delegate interfaces.

Classes

- class [AclDelegate](#)
- class [AddRoleMemberDelegate](#)
- class [BanDelegate](#)
- class [BanSuccess](#)
Convenience type that bundles all parameters passed to method [cbe::delegate::BanDelegate::onBanSuccess](#).
- class [ChunkTransferred](#)
Bundles the progress event information in connection with download and upload.
- class [CloudBackendListenerDelegate](#)
- class [CreateAccountDelegate](#)
- class [CreateContainerDelegate](#)
- class [CreateGroupDelegate](#)
- class [CreateObjectDelegate](#)
- class [CreateRoleDelegate](#)
- class [DownloadBinaryDelegate](#)
- class [DownloadBinarySuccess](#)
Convenience type that bundles all parameters passed to method [cbe::delegate::DownloadBinaryDelegate::onDownloadBinarySuccess](#).
- class [DownloadDelegate](#)

- class [DownloadSuccess](#)

Convenience type that bundles all parameters passed to method `cbe::delegate::DownloadDelegate::onDownloadSuccess`.

- class [Error](#)
- class [GetStreamsDelegate](#)
- class [GetSubscriptionsDelegate](#)
- class [ICloudBackendListener](#)
- class [LoadError](#)
- class [JoinError](#)
- class [ItemError](#)
- class [JoinDelegate](#)
- class [KickDelegate](#)
- class [KickSuccess](#)

Convenience type that bundles all parameters passed to method `cbe::delegate::KickDelegate::onKickSuccess`.

- class [LeaveDelegate](#)
- class [LeaveSuccess](#)
- class [ListGroupDelegate](#)
- class [ListMembersDelegate](#)
- class [ListRolesDelegate](#)
- class [ListSharesDelegate](#)
- class [LoginDelegate](#)
- class [PublishDelegate](#)
- class [PublishSuccess](#)

Convenience type that bundles the parameter passed to method `cbe::delegate::ShareDelegate::onShareSuccess`.

- class [QueryDelegate](#)
- class [QueryError](#)
- class [QueryJoinDelegate](#)
- struct [QueryLoaded](#)
- class [RemoveRoleDelegate](#)
- class [RemoveRoleMemberDelegate](#)
- class [SearchGroupsDelegate](#)
- class [ShareDelegate](#)
- class [ShareSuccess](#)

Convenience type that bundles the parameter passed to method `cbe::delegate::ShareDelegate::onShareSuccess`.

- class [SubscribeDelegate](#)
- class [TransferError](#)
- class [UnBanDelegate](#)
- class [UnBanSuccess](#)

Convenience type that bundles all parameters passed to method `cbe::delegate::UnBanDelegate::onUnBanSuccess`.

- class [UnPublishDelegate](#)
- class [UnPublishSuccess](#)

Convenience type that bundles all parameters passed to method `cbe::delegate::UnPublishDelegate::onUnPublishError`.

- class [UnShareDelegate](#)
- class [UnShareSuccess](#)

Convenience type that bundles the parameter passed to method `cbe::delegate::ShareDelegate::onShareSuccess`.

- class [UnSubscribeDelegate](#)
- class [UnSubscribeSuccess](#)

Convenience type that bundles all parameters passed to method `cbe::delegate::UnSubscribeDelegate::onUnSubscribeSuccess`.

- class [UpdateKeyValuesDelegate](#)
- class [UploadDelegate](#)

Typedefs

- using [AclDelegatePtr](#) = std::shared_ptr< [AclDelegate](#) >
- using [AddRoleMemberDelegatePtr](#) = std::shared_ptr< [AddRoleMemberDelegate](#) >
- using [BanDelegatePtr](#) = std::shared_ptr< [BanDelegate](#) >
- using [CloudBackendListenerDelegatePtr](#) = std::shared_ptr< [CloudBackendListenerDelegate](#) >
- using [CreateAccountDelegatePtr](#) = std::shared_ptr< [CreateAccountDelegate](#) >
- using [CreateContainerDelegatePtr](#) = std::shared_ptr< [CreateContainerDelegate](#) >
- using [CreateGroupDelegatePtr](#) = std::shared_ptr< [CreateGroupDelegate](#) >
- using [CreateObjectDelegatePtr](#) = std::shared_ptr< [CreateObjectDelegate](#) >
- using [CreateRoleDelegatePtr](#) = std::shared_ptr< [CreateRoleDelegate](#) >
- using [DownloadBinaryDelegatePtr](#) = std::shared_ptr< [DownloadBinaryDelegate](#) >
- using [DownloadDelegatePtr](#) = std::shared_ptr< [DownloadDelegate](#) >
- using [GetStreamsDelegatePtr](#) = std::shared_ptr< [GetStreamsDelegate](#) >
- using [GetSubscriptionsDelegatePtr](#) = std::shared_ptr< [GetSubscriptionsDelegate](#) >
- using [ICloudBackendListenerPtr](#) = std::shared_ptr< [ICloudBackendListener](#) >
- using [JoinDelegatePtr](#) = std::shared_ptr< [JoinDelegate](#) >
- using [KickDelegatePtr](#) = std::shared_ptr< [KickDelegate](#) >
- using [LeaveDelegatePtr](#) = std::shared_ptr< [LeaveDelegate](#) >
- using [ListGroupDelegatePtr](#) = std::shared_ptr< [ListGroupDelegate](#) >
- using [ListMembersDelegatePtr](#) = std::shared_ptr< [ListMembersDelegate](#) >
- using [ListRolesDelegatePtr](#) = std::shared_ptr< [ListRolesDelegate](#) >
- using [ListSharesDelegatePtr](#) = std::shared_ptr< [ListSharesDelegate](#) >
- using [LoginDelegatePtr](#) = std::shared_ptr< [LoginDelegate](#) >
- using [ProgressEventFn](#) = std::function< void(const [ChunkTransferred](#) &)>

Callback interface function that the [CloudBackend](#) SDK uses to indicate the progress of an upload/download.

- using [PublishDelegatePtr](#) = std::shared_ptr< [PublishDelegate](#) >
- using [QueryDelegatePtr](#) = std::shared_ptr< [QueryDelegate](#) >
- using **Error** = [delegate::Error](#)
- using [QueryJoinDelegatePtr](#) = std::shared_ptr< [QueryJoinDelegate](#) >
- using [RemoveRoleDelegatePtr](#) = std::shared_ptr< [RemoveRoleDelegate](#) >
- using [RemoveRoleMemberDelegatePtr](#) = std::shared_ptr< [RemoveRoleMemberDelegate](#) >
- using [SearchGroupsDelegatePtr](#) = std::shared_ptr< [SearchGroupsDelegate](#) >
- using [ShareDelegatePtr](#) = std::shared_ptr< [ShareDelegate](#) >
- using [SubscribeDelegatePtr](#) = std::shared_ptr< [SubscribeDelegate](#) >
- using [UnBanDelegatePtr](#) = std::shared_ptr< [UnBanDelegate](#) >
- using [UnPublishDelegatePtr](#) = std::shared_ptr< [UnPublishDelegate](#) >
- using [UnShareDelegatePtr](#) = std::shared_ptr< [UnShareDelegate](#) >
- using [UnSubscribeDelegatePtr](#) = std::shared_ptr< [UnSubscribeDelegate](#) >
- using [UpdateKeyValuesDelegatePtr](#) = std::shared_ptr< [UpdateKeyValuesDelegate](#) >
- using [UploadDelegatePtr](#) = std::shared_ptr< [UploadDelegate](#) >

Functions

- CBI::ItemDelegatePtr **createCbiQueryDelegate** ([QueryDelegatePtr](#), [cbe::util::Context](#) &&)
- CBI::ItemDelegatePtr **createCbiQueryDelegate** ([QueryJoinDelegatePtr](#), [cbe::util::Context](#) &&)
- CBI::ItemDelegatePtr **createCbiJoinDelegate** ([JoinDelegatePtr](#), [cbe::util::Context](#) &&)

11.2.1 Detailed Description

Root namespace for the delegate interfaces.

11.2.2 Typedef Documentation

11.2.2.1 AclDelegatePtr

using `cbe::delegate::AclDelegatePtr` = typedef `std::shared_ptr<AclDelegate>`

Pointer to `AclDelegate` that is passed into:

- `cbe::Container::getAcl()`
- `cbe::Container::setAcl()`
- `cbe::Object::getAcl()`
- `cbe::Object::setAcl()`

11.2.2.2 AddRoleMemberDelegatePtr

using `cbe::delegate::AddRoleMemberDelegatePtr` = typedef `std::shared_ptr<AddRoleMemberDelegate>`

Pointer to `AddRoleMemberDelegate` that is passed into:

- `cbe::Group::AddMember(AddRoleMemberDelegatePtr).`

11.2.2.3 BanDelegatePtr

using `cbe::delegate::BanDelegatePtr` = typedef `std::shared_ptr<BanDelegate>`

Pointer to `BanDelegate` that is passed into:

- `cbe::Member::ban(std::string,BanDelegatePtr)`

11.2.2.4 CloudBackendListenerDelegatePtr

using `cbe::delegate::CloudBackendListenerDelegatePtr` = typedef `std::shared_ptr<CloudBackendListenerDelegate>`

Pointer to `CloudBackendListenerDelegate` that is passed into:

`CloudBackend::addListener(CloudBackendListenerDelegatePtr).`

11.2.2.5 CreateAccountDelegatePtr

using `cbe::delegate::CreateAccountDelegatePtr` = typedef `std::shared_ptr<CreateAccountDelegate>`

Pointer to `CreateAccountDelegate` that is passed into:

`CloudBackend::createAccount(const std::string&,const std::string&,const std::string&,const std::string&,const std::string&,const std::string&,CreateAccountDelegatePtr).`

11.2.2.6 CreateContainerDelegatePtr

using `cbe::delegate::CreateContainerDelegatePtr` = typedef `std::shared_ptr<CreateContainerDelegate>`

Pointer to `CreateContainerDelegate` that is passed into:

- `cbe::Container::createContainer()`

11.2.2.7 CreateGroupDelegatePtr

using `cbe::delegate::CreateGroupDelegatePtr` = typedef `std::shared_ptr<CreateGroupDelegate>`

Pointer to `CreateGroupDelegate` that is passed into:

`Group::createGroup(std::string,std::string,CreateGroupDelegatePtr,cbe::Visibility).`

11.2.2.8 CreateObjectDelegatePtr

using `cbe::delegate::CreateObjectDelegatePtr` = typedef `std::shared_ptr<CreateObjectDelegate>`

Pointer to `CreateObjectDelegate` that is passed into:

`Container::createObject(std::string,CreateObjectDelegatePtr,cbe::KeyValues).`

11.2.2.9 CreateRoleDelegatePtr

using `cbe::delegate::CreateRoleDelegatePtr` = typedef `std::shared_ptr<CreateRoleDelegate>`

Pointer to `CreateRoleDelegate` that is passed into:

`Group::createRole(std::string, CreateRoleDelegatePtr)`.

11.2.2.10 DownloadBinaryDelegatePtr

using `cbe::delegate::DownloadBinaryDelegatePtr` = typedef `std::shared_ptr<DownloadBinaryDelegate>`

Pointer to `DownloadBinaryDelegate` that is passed into:

- `cbe::Object::download(DownloadBinaryDelegatePtr)`

11.2.2.11 DownloadDelegatePtr

using `cbe::delegate::DownloadDelegatePtr` = typedef `std::shared_ptr<DownloadDelegate>`

Pointer to `DownloadDelegate` that is passed into:

- `cbe::Object::download(const std::string&, DownloadDelegatePtr)`
- `cbe::Object::downloadStream(const std::string&, cbe::Stream, DownloadDelegatePtr)`

11.2.2.12 GetStreamsDelegatePtr

using `cbe::delegate::GetStreamsDelegatePtr` = typedef `std::shared_ptr<GetStreamsDelegate>`

Pointer to `GetStreamsDelegate` that is passed into:

`Object::getStreams(GetStreamsDelegatePtr)`.

11.2.2.13 GetSubscriptionsDelegatePtr

using `cbe::delegate::GetSubscriptionsDelegatePtr` = typedef `std::shared_ptr<GetSubscriptionsDelegate>`

Pointer to `GetSubscriptionsDelegate` that is passed into: `SubscribeManager::getSubscriptions(GetSubscriptionsDelegatePtr)`.

11.2.2.14 ICloudBackendListenerPtr

using `cbe::delegate::ICloudBackendListenerPtr` = typedef `std::shared_ptr<ICloudBackendListener>`

Pointer to `ICloudBackendListener`.

11.2.2.15 JoinDelegatePtr

typedef `std::shared_ptr<JoinDelegate>` `cbe::delegate::JoinDelegatePtr`

Pointer to `JoinDelegate` that is passed into `cbe::QueryChain::join()`.

Note

This is different from the group join.

11.2.2.16 KickDelegatePtr

using `cbe::delegate::KickDelegatePtr` = typedef `std::shared_ptr<KickDelegate>`

Pointer to `KickDelegate` that is passed into:

- `cbe::Member::kick(std::string, KickDelegatePtr)`

11.2.2.17 LeaveDelegatePtr

```
using cbe::delegate::LeaveDelegatePtr = typedef std::shared_ptr<LeaveDelegate>
```

Pointer to [LeaveDelegate](#) that is passed into:

[Group::leave\(LeaveDelegatePtr\)](#).

11.2.2.18 ListGroupsDelegatePtr

```
using cbe::delegate::ListGroupsDelegatePtr = typedef std::shared_ptr<ListGroupsDelegate>
```

Pointer to [ListGroupsDelegate](#) that is passed into:

[GroupManager::listGroups\(ListGroupsDelegatePtr\)](#).

11.2.2.19 ListMembersDelegatePtr

```
using cbe::delegate::ListMembersDelegatePtr = typedef std::shared_ptr<ListMembersDelegate>
```

Pointer to [ListMembersDelegate](#) that is passed into:

- [cbe::Group::listMembers\(ListMembersDelegatePtr\)](#).

11.2.2.20 ListRolesDelegatePtr

```
using cbe::delegate::ListRolesDelegatePtr = typedef std::shared_ptr<ListRolesDelegate>
```

Pointer to [ListRolesDelegate](#) that is passed into:

- [cbe::Group::ListRoles\(ListRolesDelegatePtr\)](#).

11.2.2.21 ListSharesDelegatePtr

```
using cbe::delegate::ListSharesDelegatePtr = typedef std::shared_ptr<ListSharesDelegate>
```

Pointer to [ListSharesDelegate](#) that is passed into:

- [ShareManager::listAvailableShares\(ListSharesDelegatePtr\)](#)
- [ShareManager::listMyShares\(ListSharesDelegatePtr\)](#)

11.2.2.22 LoginDelegatePtr

```
using cbe::delegate::LoginDelegatePtr = typedef std::shared_ptr<LoginDelegate>
```

Pointer to [LoginDelegate](#) that is passed into:

- [cbe::CloudBackend::login\(const std::string&,const std::string&,const std::string&,LoginDelegatePtr\)](#)

11.2.2.23 ProgressEventFn

```
using cbe::delegate::ProgressEventFn = typedef std::function<void(const ChunkTransferred&)>
```

Callback interface function that the [CloudBackend](#) SDK uses to indicate the progress of an upload/download.

11.2.2.24 PublishDelegatePtr

```
using cbe::delegate::PublishDelegatePtr = typedef std::shared_ptr<PublishDelegate>
```

Pointer to [PublishDelegate](#) that is passed into:

- [cbe::Container::publish\(\)](#)
- [cbe::Object::publish\(\)](#)

11.2.2.25 QueryDelegatePtr

```
typedef std::shared_ptr< QueryDelegate > cbe::delegate::QueryDelegatePtr
```

Pointer to [QueryDelegate](#) that is passed into:

- [CloudBackend::query\(ContainerId,delegate::QueryDelegatePtr\)](#) and overloads.
- [Container::query\(delegate::QueryDelegatePtr\)](#) and overloads.

11.2.2.26 QueryJoinDelegatePtr

```
typedef std::shared_ptr< QueryJoinDelegate > cbe::delegate::QueryJoinDelegatePtr
```

Pointer to [QueryJoinDelegate](#) that is passed into:

- [CloudBackend::query\(ContainerId,delegate::QueryJoinDelegatePtr\)](#) and overloads.
- [Container::query\(delegate::QueryJoinDelegatePtr\)](#) and overloads.

11.2.2.27 RemoveRoleDelegatePtr

```
using cbe::delegate::RemoveRoleDelegatePtr = typedef std::shared_ptr<RemoveRoleDelegate>
```

Pointer to [RemoveRoleDelegate](#) that is passed into:

[Group::removeRole\(std::string, RemoveRoleDelegatePtr\)](#).

11.2.2.28 RemoveRoleMemberDelegatePtr

```
using cbe::delegate::RemoveRoleMemberDelegatePtr = typedef std::shared_ptr<RemoveRoleMemberDelegate>
```

Pointer to [RemoveRoleMemberDelegate](#) that is passed into:

- [cbe::Group::RemoveMember\(RemoveRoleMemberDelegatePtr\)](#).

11.2.2.29 SearchGroupsDelegatePtr

```
using cbe::delegate::SearchGroupsDelegatePtr = typedef std::shared_ptr<SearchGroupsDelegate>
```

Pointer to [SearchGroupsDelegate](#) that is passed into:

- [cbe::GroupManager::searchGroups\(\)](#)

11.2.2.30 ShareDelegatePtr

```
using cbe::delegate::ShareDelegatePtr = typedef std::shared_ptr<ShareDelegate>
```

Pointer to [ShareDelegate](#) that is passed into:

- [cbe::Container::share\(cbe::UserId,std::string,ShareDelegatePtr\)](#)
- [cbe::Object::share\(cbe::UserId,std::string,ShareDelegatePtr\)](#)

11.2.2.31 SubscribeDelegatePtr

```
using cbe::delegate::SubscribeDelegatePtr = typedef std::shared_ptr<SubscribeDelegate>
```

Pointer to [SubscribeDelegate](#) that is passed into:

- [cbe::SubscribeManager::subscribe\(cbe::UserId,std::string,cbe::PublishId,std::string,std::string,std::string,SubscribeDelegatePtr\)](#)

11.2.2.32 UnBanDelegatePtr

using `cbe::delegate::UnBanDelegatePtr` = typedef `std::shared_ptr<UnBanDelegate>`

Pointer to `UnBanDelegate` that is passed into:

- `cbe::Member::unBan(UnBanDelegatePtr)`

11.2.2.33 UnPublishDelegatePtr

using `cbe::delegate::UnPublishDelegatePtr` = typedef `std::shared_ptr<UnPublishDelegate>`

Pointer to `UnPublishDelegate` that is passed into:

- `Container::unPublish(UnPublishDelegatePtr)`
- `Object::unPublish(UnPublishDelegatePtr)`

11.2.2.34 UnShareDelegatePtr

using `cbe::delegate::UnShareDelegatePtr` = typedef `std::shared_ptr<UnShareDelegate>`

Pointer to `UnShareDelegate` that is passed into:

- `cbe::Container::unShare()`
- `cbe::Object::unShare()`

11.2.2.35 UnSubscribeDelegatePtr

using `cbe::delegate::UnSubscribeDelegatePtr` = typedef `std::shared_ptr<UnSubscribeDelegate>`

Pointer to `UnSubscribeDelegate` that is passed into:

- `Container::unSubscribe(UnSubscribeDelegatePtr)`
- `Object::unSubscribe(UnSubscribeDelegatePtr)`

11.2.2.36 UpdateKeyValuesDelegatePtr

using `cbe::delegate::UpdateKeyValuesDelegatePtr` = typedef `std::shared_ptr<UpdateKeyValuesDelegate>`

Pointer to `UpdateKeyValuesDelegate` that is passed into:

- `cbe::Object::updateKeyValues()`

11.2.2.37 UploadDelegatePtr

using `cbe::delegate::UploadDelegatePtr` = typedef `std::shared_ptr<UploadDelegate>`

Pointer to `UploadDelegate` that is passed into:

- `cbe::Container::upload(const std::string&, UploadDelegatePtr)`
- `cbe::Container::upload(const std::string&, const std::string&, UploadDelegatePtr)`
- `cbe::Object::uploadStream(const std::string&, cbe::StreamId, UploadDelegatePtr)`

11.3 cbe::delegate::container Namespace Reference

Namespace for `cbe::Container` specific delegate interfaces.

Classes

- class [MoveDelegate](#)
- class [RemoveDelegate](#)
- class [RemoveSuccess](#)

Convenience type that bundles all parameters passed to method [cbe::delegate::container::RemoveDelegate::onRemoveSuccess](#).

- class [RenameDelegate](#)

Typedefs

- using [MoveDelegatePtr](#) = std::shared_ptr< [MoveDelegate](#) >
- using [RemoveDelegatePtr](#) = std::shared_ptr< [RemoveDelegate](#) >
- using [RenameDelegatePtr](#) = std::shared_ptr< [RenameDelegate](#) >

11.3.1 Detailed Description

Namespace for [cbe::Container](#) specific delegate interfaces.

11.3.2 Typedef Documentation

11.3.2.1 MoveDelegatePtr

using [cbe::delegate::container::MoveDelegatePtr](#) = typedef std::shared_ptr<[MoveDelegate](#)>
 Pointer to [MoveDelegate](#) that is passed into: [Container::move](#)(void move([cbe::ContainerId](#),[MoveDelegatePtr](#)).

11.3.2.2 RemoveDelegatePtr

using [cbe::delegate::container::RemoveDelegatePtr](#) = typedef std::shared_ptr<[RemoveDelegate](#)>
 Pointer to [RemoveDelegate](#) that is passed into:

- [cbe::Container::remove](#)([RemoveDelegatePtr](#))

11.3.2.3 RenameDelegatePtr

using [cbe::delegate::container::RenameDelegatePtr](#) = typedef std::shared_ptr<[RenameDelegate](#)>
 Pointer to [RenameDelegate](#) that is passed into: [Container::rename](#)(const std::string&,[RenameDelegatePtr](#)).

11.4 cbe::delegate::group Namespace Reference

Namespace for [cbe::Group](#) specific delegate interfaces.

Classes

- class [JoinDelegate](#)
- class [RemoveDelegate](#)
- class [RemoveSuccess](#)

Convenience type that bundles all parameters passed to method [cbe::delegate::group::RemoveDelegate::onRemoveSuccess](#).

- class [RenameDelegate](#)
- class [RenameSuccess](#)

Convenience type that bundles all parameters passed to method [cbe::delegate::group::RenameDelegate::onRenameSuccess](#).

Typedefs

- using [JoinDelegatePtr](#) = std::shared_ptr< [JoinDelegate](#) >
- using [RemoveDelegatePtr](#) = std::shared_ptr< [RemoveDelegate](#) >
- using [RenameDelegatePtr](#) = std::shared_ptr< [RenameDelegate](#) >

11.4.1 Detailed Description

Namespace for [cbe::Group](#) specific delegate interfaces.

11.4.2 Typedef Documentation

11.4.2.1 JoinDelegatePtr

using [cbe::delegate::group::JoinDelegatePtr](#) = typedef std::shared_ptr<[JoinDelegate](#)>
 Pointer to [JoinDelegate](#) that is passed into: [cbe::Group::join\(\)](#)

Note

This is different from the query chain join.

11.4.2.2 RemoveDelegatePtr

using [cbe::delegate::group::RemoveDelegatePtr](#) = typedef std::shared_ptr<[RemoveDelegate](#)>
 Pointer to [RemoveDelegate](#) that is passed into:

- [cbe::Group::remove\(RemoveDelegatePtr\)](#)

11.4.2.3 RenameDelegatePtr

using [cbe::delegate::group::RenameDelegatePtr](#) = typedef std::shared_ptr<[RenameDelegate](#)>
 Pointer to [RenameDelegate](#) that is passed into: [cbe::Group::rename\(\)](#)

11.5 cbe::delegate::object Namespace Reference

Namespace for [cbe::Object](#) specific delegate interfaces.

Classes

- class [MoveDelegate](#)
- class [RemoveDelegate](#)
- class [RemoveSuccess](#)
Convenience type that bundles all parameters passed to method [cbe::delegate::object::onRemoveSuccess](#).
- class [RenameDelegate](#)

Typedefs

- using [MoveDelegatePtr](#) = std::shared_ptr< [MoveDelegate](#) >
- using [RemoveDelegatePtr](#) = std::shared_ptr< [RemoveDelegate](#) >
- using [RenameDelegatePtr](#) = std::shared_ptr< [RenameDelegate](#) >

11.5.1 Detailed Description

Namespace for [cbe::Object](#) specific delegate interfaces.

11.5.2 Typedef Documentation

11.5.2.1 MoveDelegatePtr

using `cbe::delegate::object::MoveDelegatePtr` = typedef `std::shared_ptr<MoveDelegate>`
 Pointer to `MoveDelegate` that is passed into: `cbe::Object::move()`

11.5.2.2 RemoveDelegatePtr

using `cbe::delegate::object::RemoveDelegatePtr` = typedef `std::shared_ptr<RemoveDelegate>`
 Pointer to `RemoveDelegate` that is passed into:

- `cbe::Object::remove(RemoveDelegatePtr)`

11.5.2.3 RenameDelegatePtr

using `cbe::delegate::object::RenameDelegatePtr` = typedef `std::shared_ptr<RenameDelegate>`
 Pointer to `RenameDelegate` that is passed into: `cbe::Object::rename()`

11.6 cbe::util Namespace Reference

General support program utilities.

Classes

- struct `Context`
- struct `ErrorInfo`
- struct `ErrorInfoImpl`
- struct `Exception`
- struct `ExceptionImpl`
- class `Optional`

Class template `Optional` manages an optional contained value - i.e., a value that is either present or absent.

Functions

- template<typename T, typename... Ts>
`std::ostream & buildMsg` (`std::ostream &os`, `T &&arg`, `Ts &&... restArgs`)
- template<typename T, typename... Ts>
`std::enable_if<!std::is_base_of< std::ostream, typename std::remove_reference< T >::type >::value, std::string >::type` `buildMsg` (`T &&arg`, `Ts &&... restArgs`)
- `std::ostream & operator<<` (`std::ostream &os`, const `ErrorInfo &ei`)

11.6.1 Detailed Description

General support program utilities.

Chapter 12

Class Documentation

12.1 cbe::Account Class Reference

Login account information.

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

Public Member Functions

- `cbe::UserId` `userId` () const
Returns the account id of the user.
- `std::string` `username` () const
Returns the username of the account.
- `std::string` `password` () const
Returns the password of the account.
- `std::string` `source` () const
Returns the tenant name of the account.
- `std::string` `client` () const
Returns the client of the account.
- `std::string` `firstName` () const
Returns the given name of the user.
- `std::string` `lastName` () const
Returns the surname of the user.
- `cbe::Container` `rootContainer` () const
Returns the rootContainer for the account.
- `cbe::ContainerId` `tenantContainerId` () const
Returns the tenant ContainerId.
- `cbe::ContainerId` `libContainerId` () const
Returns the library ContainerId.
- `cbe::DatabaseId` `rootDatabase` () const
Returns the DatabaseId for home://.
- `cbe::DataBases` `databases` () const
Returns the Databases available.
- `Account` (`cbe::DefaultCtor`)
Default constructor.
- `operator bool` () const
Checks if the current instance is real.

Friends

- class `CloudBackend`

12.1.1 Detailed Description

Login account information.

12.1.2 Constructor & Destructor Documentation

12.1.2.1 Account()

```
cbe::Account::Account (
    cbe::DefaultCtor )
```

Default constructor.

Construct a new object with the `DefaultCtor` to enable the `operator bool()` test

12.1.3 Member Function Documentation

12.1.3.1 client()

```
std::string cbe::Account::client ( ) const
```

Returns the client of the account.

If it was set when the account was created.

12.1.3.2 databases()

```
cbe::DataBases cbe::Account::databases ( ) const
```

Returns the Databases available.

Returns a `std::map` of databases available for the account.

12.1.3.3 firstName()

```
std::string cbe::Account::firstName ( ) const
```

Returns the given name of the user.

Stored in lowercase.

12.1.3.4 libContainerId()

```
cbe::ContainerId cbe::Account::libContainerId ( ) const
```

Returns the library `ContainerId`.

The `ContainerId` of the library database for the account, which is where application specific settings and application state can be permanently stored in order to be available next time the app starts.

12.1.3.5 operator bool()

```
cbe::Account::operator bool ( ) const [explicit]
```

Checks if the current instance is real.

An "unreal" instance implies typically a failed event.

Relies on the `Default` constructor [Account\(cbe::DefaultCtor\)](#)

Returns

`true` : is real

`false` : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

12.1.3.6 password()

```
std::string cbe::Account::password ( ) const
```

Returns the password of the account.

What was used to login.

12.1.3.7 rootContainer()

```
cbe::Container cbe::Account::rootContainer ( ) const
```

Returns the rootContainer for the account.

The top container is also known as home://

12.1.3.8 rootDatabase()

```
cbe::DatabaseId cbe::Account::rootDatabase ( ) const
```

Returns the DatabaseId for home://.

Returns the databaseId for root a.k.a. home:// of the account.

12.1.3.9 source()

```
std::string cbe::Account::source ( ) const
```

Returns the tenant name of the account.

Note

Stored in lowercase.

12.1.3.10 tenantContainerId()

```
cbe::ContainerId cbe::Account::tenantContainerId ( ) const
```

Returns the tenant ContainerId.

Returns the ContainerId of the tenant database for the account. The top container is also known as tenant://

To get the [Container](#), see [Databases](#)

12.1.3.11 username()

```
std::string cbe::Account::username ( ) const
```

Returns the username of the account.

Stored in lowercase.

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

- include/cbe/Account.h

12.2 cbe::delegate::AcIDelegate Class Reference

```
#include <AcIDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [AcIMap](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onAclSuccess](#) ([AclMap](#) &&aclMap)=0
- virtual void [onAclError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.2.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Container::getAcl\(\)](#)
- [cbe::Container::setAcl\(\)](#)
- [cbe::Object::getAcl\(\)](#)
- [cbe::Object::setAcl\(\)](#)

12.2.2 Member Function Documentation

12.2.2.1 onAclError()

```
virtual void cbe::delegate::AclDelegate::onAclError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.2.2.2 onAclSuccess()

```
virtual void cbe::delegate::AclDelegate::onAclSuccess (
    AclMap && aclMap ) [pure virtual]
```

Called upon successful GetACL.

Parameters

aclMap	The permissions for requested cbe::Container or cbe::Object .
------------------------	---

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

- include/cbe/delegate/AclDelegate.h

12.3 cbe::delegate::AddRoleMemberDelegate Class Reference

```
#include <AddRoleMemberDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [MemberId](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onAddRoleMemberSuccess](#) ([MemberId](#) memberId)=0
- virtual void [onAddRoleMemberError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.3.1 Detailed Description

Delegate class for the asynchronous version of method:

- cbe::Group::AddMember

12.3.2 Member Function Documentation

12.3.2.1 onAddRoleMemberError()

```
virtual void cbe::delegate::AddRoleMemberDelegate::onAddRoleMemberError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.3.2.2 onAddRoleMemberSuccess()

```
virtual void cbe::delegate::AddRoleMemberDelegate::onAddRoleMemberSuccess (
    MemberId memberId ) [pure virtual]
```

Called upon successful AddMember.

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

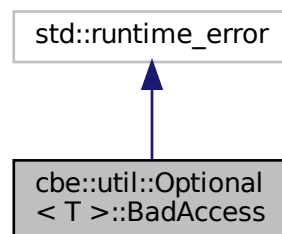
- include/cbe/delegate/AddRoleMemberDelegate.h

12.4 cbe::util::Optional< T >::BadAccess Class Reference

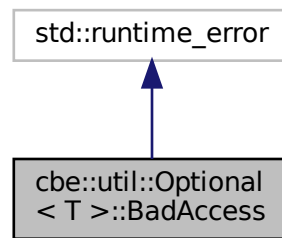
Thrown by [value\(\)](#) method when accessing an object that does not contain a value.

```
#include <Optional.h>
```

Inheritance diagram for cbe::util::Optional< T >::BadAccess:



Collaboration diagram for `cbe::util::Optional< T >::BadAccess`:



12.4.1 Detailed Description

```
template<typename T>
class cbe::util::Optional< T >::BadAccess
```

Thrown by [value\(\)](#) method when accessing an object that does not contain a value.
The documentation for this class was generated from the following file:

- `include/cbe/util/Optional.h`

12.5 cbe::delegate::BanDelegate Class Reference

```
#include <BanDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [BanSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onBanSuccess](#) (std::string &&memberName, [cbe::MemberId](#) memberId)=0
- virtual void [onBanError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.5.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Member::ban](#)

12.5.2 Member Function Documentation

12.5.2.1 onBanError()

```
virtual void cbe::delegate::BanDelegate::onBanError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.5.2.2 onBanSuccess()

```
virtual void cbe::delegate::BanDelegate::onBanSuccess (
    std::string && memberName,
    cbe::MemberId memberId ) [pure virtual]
```

Called upon successful ban.

Parameters

<i>memberName</i>	Name of the member being banned.
<i>memberId</i>	Id of the member being banned.

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

- include/cbe/delegate/BanDelegate.h

12.6 cbe::delegate::BanSuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::BanDelegate::onBanSuccess](#).

```
#include <BanDelegate.h>
```

Public Member Functions

- **BanSuccess** ([cbe::DefaultCtor](#))
- **BanSuccess** (std::string &&memberName, [cbe::MemberId](#) memberId)

Public Attributes

- std::string **memberName** {}
- [cbe::MemberId](#) **memberId** {}

12.6.1 Detailed Description

Convenience type that bundles all parameters passed to method [cbe::delegate::BanDelegate::onBanSuccess](#).

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

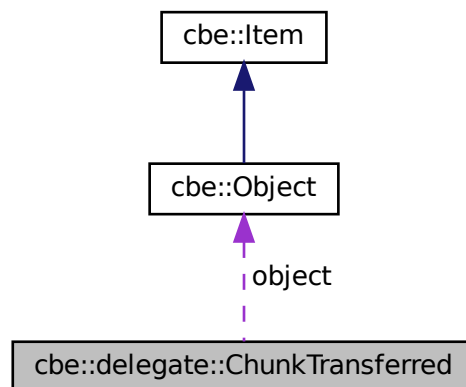
- include/cbe/delegate/BanDelegate.h

12.7 cbe::delegate::ChunkTransferred Class Reference

Bundles the progress event information in connection with download and upload.

```
#include <ChunkTransferred.h>
```

Collaboration diagram for `cbe::delegate::ChunkTransferred`:



Public Member Functions

- **ChunkTransferred** ([cbe::DefaultCtor](#))
- **ChunkTransferred** ([cbe::Object](#) &&object, std::uint64_t transferred, std::uint64_t total)

Public Attributes

- [cbe::Object](#) **object** {[cbe::DefaultCtor](#){}}
- std::uint64_t **transferred** {}
- std::uint64_t **total** {}

12.7.1 Detailed Description

Bundles the progress event information in connection with download and upload.
Convenience type that bundles all parameters passed to methods:

- [cbe::delegate::DownloadDelegate::onChunkReceived\(\)](#)
- [cbe::delegate::DownloadBinaryDelegate::onChunkReceived\(\)](#)
- [cbe::delegate::UploadDelegate::onChunkSent\(\)](#)
- [cbe::delegate::UploadBinaryDelegate::onChunkSent\(\)](#)

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

- `include/cbe/delegate/ChunkTransferred.h`

12.8 cbe::CloudBackend Class Reference

The session that holds the connection with the cloud.

```
#include <CloudBackend.h>
```

Public Types

- using `LoginDelegatePtr` = `delegate::LoginDelegatePtr`
- using `CreateAccountDelegatePtr` = `delegate::CreateAccountDelegatePtr`
- using `CloudBackendListenerDelegatePtr` = `std::shared_ptr< delegate::CloudBackendListenerDelegate >`
- using `ListenerHandle` = `std::uint64_t`
Handle identifying a registered Listener.
- using `QueryDelegatePtr` = `delegate::QueryDelegatePtr`
- using `QueryJoinDelegatePtr` = `delegate::QueryJoinDelegatePtr`

Public Member Functions

- `ListenerHandle addListener (CloudBackendListenerDelegatePtr listener)`
Listen for changes to specific items.
- `void removeListener (ListenerHandle handle)`
Delete a specific listener.
- `cbe::QueryChain query (ContainerId containerId, QueryDelegatePtr queryDelegate)`
Select Item's of a container (table).
- `cbe::QueryChain query (ContainerId containerId, Filter filter, QueryDelegatePtr queryDelegate)`
Select Item's of a container (table) with a filter (where).
- `cbe::QueryChainExt query (ContainerId containerId, QueryJoinDelegatePtr queryJoinDelegate)`
Join multiple tables.
- `cbe::QueryChainExt query (ContainerId containerId, Filter filter, QueryJoinDelegatePtr queryJoinDelegate)`
Join multiple tables using filter (where).
- `cbe::QueryChain queryWithPath (std::string relativePath, cbe::ContainerId queryRoot, QueryDelegatePtr delegate)`
Queries items of a container with a given path.
- `cbe::QueryChain queryWithPath (std::string relativePath, QueryDelegatePtr delegate)`
Queries items of a container with a given path relative the home root container.
- `cbe::QueryResult search (std::string tags, cbe::ContainerId containerId, QueryDelegatePtr delegate)`
Select Object's with specified key/values.
- `cbe::QueryResult search (cbe::Filter filter, cbe::ContainerId containerId, QueryDelegatePtr delegate)`
Select Object's with specified key/values using filter.
- `bool clearCache ()`
Clear the local cache.
- `cbe::Account account ()`
Returns an account object with information on the user.
- `std::string version () const`
Returns the version number of the SDK.
- `cbe::GroupManager groupManager ()`
Gets the group manager.
- `cbe::ShareManager shareManager ()`
Gets the share manager.
- `cbe::PublishManager publishManager ()`
Gets the publish manager.
- `void terminate ()`
Terminates the CloudBackend service.
- `cbe::SubscribeManager subscribeManager ()`
Gets the subscribe manager.
- `CloudBackend (cbe::DefaultCtor)`
Default constructor.
- `operator bool () const`
Checks if the current instance is real.

Static Public Member Functions

- static [cbe::CloudBackend login](#) (const std::string &username, const std::string &password, const std::string &tenant, const std::string &client, [LoginDelegatePtr](#) delegate)
Logs in to the CloudBackend service.
- static [cbe::UserId createAccount](#) (const std::string &username, const std::string &password, const std::string &email, const std::string &firstName, const std::string &lastName, const std::string &tenant, const std::string &client, [CreateAccountDelegatePtr](#) delegate)
Creates a user account.
- static [cbe::Container castContainer](#) ([cbe::Item](#) item)
Casts an item to a container.
- static [cbe::Object castObject](#) ([cbe::Item](#) item)
Casts an item to an object.

12.8.1 Detailed Description

The session that holds the connection with the cloud.
It ensures the connection is authenticated.

12.8.2 Member Typedef Documentation

12.8.2.1 CloudBackendListenerDelegatePtr

using [cbe::CloudBackend::CloudBackendListenerDelegatePtr](#) = std::shared_ptr<[delegate::CloudBackendListenerDelegate](#)>
Pointer to [delegate::CloudBackendListenerDelegate](#) that is passed into asynchronous version of method [addListener\(CloudBackendListenerDelegatePtr\)](#)

12.8.2.2 CreateAccountDelegatePtr

using [cbe::CloudBackend::CreateAccountDelegatePtr](#) = [delegate::CreateAccountDelegatePtr](#)
Pointer to [delegate::CreateAccountDelegate](#) that is passed into asynchronous version of method [createAccount\(\)](#)

12.8.2.3 LoginDelegatePtr

using [cbe::CloudBackend::LoginDelegatePtr](#) = [delegate::LoginDelegatePtr](#)
Pointer to [cbe::delegate::LoginDelegate\(\)](#) that is passed into asynchronous version of method [login\(\)](#)

12.8.2.4 QueryDelegatePtr

using [cbe::CloudBackend::QueryDelegatePtr](#) = [delegate::QueryDelegatePtr](#)
Pointer to [delegate::QueryDelegate](#) that is passed into asynchronous version of methods:

- [query\(ContainerId,QueryDelegatePtr\)](#),
- [query\(ContainerId,Filter,QueryDelegatePtr\)](#),
- [queryWithPath\(std::string, QueryDelegatePtr\)](#)
- [queryWithPath\(std::string, cbe::ContainerId, QueryDelegatePtr\)](#),
- [search\(std::string,cbe::ContainerId,QueryDelegatePtr\)](#),
- [search\(Filter,cbe::ContainerId,QueryDelegatePtr\)](#)

12.8.2.5 QueryJoinDelegatePtr

using `cbe::CloudBackend::QueryJoinDelegatePtr = delegate::QueryJoinDelegatePtr`
 Pointer to `delegate::QueryJoinDelegate` that is passed into asynchronous version of methods

- `query(ContainerId,QueryJoinDelegatePtr)`
- `query(ContainerId,Filter,QueryJoinDelegatePtr)`

12.8.3 Constructor & Destructor Documentation

12.8.3.1 CloudBackend()

```
cbe::CloudBackend::CloudBackend (
    cbe::DefaultCtor )
```

Default constructor.

Construct a new object with the `DefaultCtor` to enable the `operator bool()` test

12.8.4 Member Function Documentation

12.8.4.1 account()

```
cbe::Account cbe::CloudBackend::account ( )
```

Returns an account object with information on the user.

Returns

`cbe::Account`

12.8.4.2 addListener()

```
ListenerHandle cbe::CloudBackend::addListener (
    CloudBackendListenerDelegatePtr listener )
```

Listen for changes to specific items.

Adds a listener that will receive updates as changes occur on the account.

Note

`removeListener()` should always be called when no longer using the delegate.

Parameters

<i>listener</i>	Delegate is a shared pointer to the class <code>delegate::CloudBackendListenerDelegate</code> that the user has implemented
-----------------	---

Returns

Handle identifying the registration of the listener.

To be used when de-registering with method `removeListener(ListenerHandle)`.

12.8.4.3 castContainer()

```
static cbe::Container cbe::CloudBackend::castContainer (
```

```
cbe::Item item ) [static]
```

Casts an item to a container.

Pointer to [cbe::delegate::QueryDelegate](#) that is passed into asynchronous version of method `search()`

If the provided [cbe::Item](#) "item" is not a [cbe::Container](#) "container", an empty container instance is returned

12.8.4.4 castObject()

```
static cbe::Object cbe::CloudBackend::castObject (
    cbe::Item item ) [static]
```

Casts an item to an object.

If the provided [cbe::Item](#) "item" is not a [cbe::Object](#) "object", an empty object instance is returned

12.8.4.5 clearCache()

```
bool cbe::CloudBackend::clearCache ( )
```

Clear the local cache.

Use if there is a local SDK memory issue.

Returns

true success

false failed

12.8.4.6 createAccount()

```
static cbe::UserId cbe::CloudBackend::createAccount (
    const std::string & username,
    const std::string & password,
    const std::string & email,
    const std::string & firstName,
    const std::string & lastName,
    const std::string & tenant,
    const std::string & client,
    CreateAccountDelegatePtr delegate ) [static]
```

Creates a user account.

This method is only granted to the tenant administrator.

Asynchronous version of this service function.

Parameters

<i>username</i>	Username for the new account
<i>password</i>	Password for the new account
<i>email</i>	Email address to owner of the new account
<i>firstName</i>	First name of the owner of the new account
<i>lastName</i>	Last name of the owner of the new account
<i>tenant</i>	The identifier for the tenant, formerly known as <code>source</code> .
<i>client</i>	Describing platform running the software. Names defined by the tenant administrator.
<i>delegate</i>	Pointer to a delegate::CreateAccountDelegate instance that is implemented by the user.

12.8.4.7 groupManager()

```
cbe::GroupManager cbe::CloudBackend::groupManager ( )
```

Gets the group manager.

Returns

[cbe::GroupManager](#)

12.8.4.8 logIn()

```
static cbe::CloudBackend cbe::CloudBackend::logIn (
    const std::string & username,
    const std::string & password,
    const std::string & tenant,
    const std::string & client,
    LoginDelegatePtr delegate ) [static]
```

Logs in to the CloudBackend service.

When finished with the usage of the CloudBackend service, call method [CloudBackend::terminate\(\)](#) in all cases.

The authentication to the account is determined by username, password and tenant.

Asynchronous version of this service function.

Parameters

<i>username</i>	Name of the user to be signed in.
<i>password</i>	Password for the user to be signed in.
<i>tenant</i>	The identifier for the tenant, formerly known as <code>source</code> .
<i>client</i>	Identifier of what type of client the program is running on. This is used by the tenant for statistics and selective communication with the users. The tenant administrator defines what client names to use.
<i>delegate</i>	Pointer to a <code>be::CloudBackend::LoginDelegate()</code> instance that is implemented by the user to receive the response of this login request. This is accomplished in terms of the LoginDelegate callback functions onLoginSuccess() and onLoginError() .

Example

Async login

```
#include "cbe/CloudBackend.h"
~~~
#include <condition_variable>
#include <mutex>
~~~
int main() {
    // First, declare and implement a LoginDelegate class
    class MyLoginDelegate : public cbe::delegate::LoginDelegate {
    public:
        std::mutex mutex{};
        std::condition_variable conditionVariable{};
        // Indicates operation completed - success or failure
        bool called{};
    private:
        // Default construct this CloudBackend member. If the method
        // onLoginSuccess() will not be called, this default constructed state
        // implies that this cloudBackend object is not valid due to a successful
        // log in
        cbe::CloudBackend cloudBackend{ cbe::DefaultCtor() };
        // Default construct this cbe::delegate::Error member, If the method
        // onLoginError() will not be called, this default constructed state
        // implies no error. Otherwise, thus error object will contain the error
        // information.
        cbe::delegate::Error error{};
        // Implement the cbe::delegate::LoginDelegate interface callbacks (private
        // cause they are only used by the SDK):
    private:
        // Callback called upon successful login.
        void onLoginSuccess(cbe::CloudBackend&& cloudBackend) final {
        {
            std::lock_guard<std::mutex> lock(mutex);
            // Put error member into the default constructed state to
            // indicate success (i.e., no error)
            error = cbe::delegate::Error();
            // Change state of cloudBackend member to indicate success
            this->cloudBackend = std::move(cloudBackend);
```

```

        // Indicate operation completed, member cloudBackend non-default
        // constructed state indicates success
        called = true;
    }
    conditionVariable.notify_one();
}
// Callback called upon failed login.
void onLoginError(Error&& error, cbe::util::Context&& context) final {
    {
        std::lock_guard<std::mutex> lock(mutex);
        // Put cloudBackend member into the default constructed state to
        // indicate no-success
        cloudBackend = cbe::CloudBackend{ cbe::DefaultCtor{} };
        // Change state of error member to indicate failure
        this->error = std::move(error);
        // Indicate operation completed, member member cloudBackend default
        // constructed state indicates failure
        called = true;
    }
    conditionVariable.notify_one();
}
// -----
public:
    void waitForRsp() {
        std::unique_lock<std::mutex> lock(mutex);
        conditionVariable.wait(lock, [this]{ return called; });
        // Reset called flag, so current delegate instance can be reused
        called = false;
    }
}; // struct MyLoginDelegate
~~~
~~~
// Second, initiate the login:
static constexpr const char* const username = "~~~";
static constexpr const char* const password = "~~~";
static constexpr const char* const tenant   = "~~~";
static constexpr const char* const client   = "linux_desktop";
~~~
std::shared_ptr<MyLoginDelegate> myLoginDelegate =
                                std::make_shared<MyLoginDelegate>();
// Invoke the login (cast to return value to void to indicate is deliberately
// discarded)
cbe::CloudBackend cloudBackend = cbe::CloudBackend::login(username, password, tenant, client,
myLoginDelegate);
// Await the login procedure to run to completion
myLoginDelegate->waitForRsp();
// Test if login failed
if (myLoginDelegate->error) {
    std::cout << "Failed to login: error={" << myLoginDelegate->error << '}'
               << std::endl;
    // Bail out!
    ~~~
}
// Since we have a complete CloudBackend object now, copy it from the delegate
cloudBackend = myLoginDelegate->cloudBackend;
// Dispose the delegate object, not needed anymore
myLoginDelegate.reset();
// Do your stuff
~~~
// End of program, tell cbe to terminate the session
cloudBackend.terminate();
}

```

12.8.4.9 operator bool()

cbe::CloudBackend::operator bool () const [explicit]

Checks if the current instance is real.

An "unreal" instance implies typically a failed login event.

Relies on the Default constructor [CloudBackend\(cbe::DefaultCtor\)](#)

Returns

true : is real

false : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

12.8.4.10 publishManager()

```
cbe::PublishManager cbe::CloudBackend::publishManager ( )
```

Gets the publish manager.

Returns

[cbe::PublishManager](#)

12.8.4.11 query() [1/4]

```
cbe::QueryChain cbe::CloudBackend::query (
    ContainerId containerId,
    Filter filter,
    QueryDelegatePtr queryDelegate )
```

Select Item's of a container (table) with a filter (where).

Same as [query\(ContainerId,QueryDelegatePtr\)](#), but with an additional parameter *filter*.

Parameters

<i>filter</i>	Filter specifying the constraints of the requested items.
---------------	---

12.8.4.12 query() [2/4]

```
cbe::QueryChainExt cbe::CloudBackend::query (
    ContainerId containerId,
    Filter filter,
    QueryJoinDelegatePtr queryJoinDelegate )
```

Join multiple tables using filter (where).

Same as [query\(ContainerId,delegate::QueryJoinDelegatePtr\)](#), but with an additional [Filter](#) parameter *filter*.

Parameters

<i>filter</i>	Filter specifying the constraints of the requested items.
---------------	---

12.8.4.13 query() [3/4]

```
cbe::QueryChain cbe::CloudBackend::query (
    ContainerId containerId,
    QueryDelegatePtr queryDelegate )
```

Select Item's of a container (table).

Inquires for a set of [Items](#) from the provided Container, identified by *containerId*. Response is delivered asynchronously via the [delegate::QueryDelegate](#) callback interface passed in as argument via parameter *queryDelegate*.

This overload of *join()* accepting a [QueryDelegate](#)-pointer as parameter has two use cases:

1. To make a solely [query\(\)](#) call **without** an additional chained [join\(\)](#)-call.
2. To make a [query\(\)](#) call **with** a desired chained [join\(\)](#) call.
In this latter case, the overloaded [query\(\)](#)-functions:

- [query\(ContainerId,delegate::QueryJoinDelegatePtr\)](#)
- [query\(ContainerId,Filter,delegate::QueryJoinDelegatePtr\)](#)

both accepting a [QueryJoinDelegate](#) as *delegate* argument, must be used.

Asynchronous version of this service function.

Parameters

<i>containerId</i>	Id of the container which contents will inquired.
<i>queryDelegate</i>	Pointer to a QueryDelegate instance, implemented by the user, that receives the response of this query request. I.e., either the QueryDelegate callback function onQuerySuccess() or onQueryError() will be called in the event of success, or failure respectively.

Example

Async query

```

~~~
std::shared_ptr<MyQueryDelegate> queryDelegate =
    std::make_shared<MyQueryDelegate>();
const cbe::ContainerId myContainerId =
    cloudBackend.account().rootContainer().id();
cloudBackend.query(myContainerId, queryDelegate);
queryDelegate->waitForRsp();
if (queryDelegate->errorInfo) {
    std::cout << "Query failed:" << std::endl;
    std::cout << "ErrorInfo = " << queryDelegate->errorInfo << std::endl;
} else {
    cbe::QueryResult::ItemsSnapshot itemsSnapshot =
        queryDelegate->queryResult.getItemsSnapshot();
    for (auto& item : itemsSnapshot) {
        std::cout << item.name() << std::endl;
    }
}
~~~

```

To use the code above, we must first login and then declare a [QueryDelegate](#) class.

See also:

- [Async login](#)
Here, we retrieve the `cloudBackend` object.
- [Example QueryDelegatePtr of a QueryDelegate implementation class](#)
Here, class `MyQueryDelegate` is defined.

12.8.4.14 query() [4/4]

```

cbe::QueryChainExt cbe::CloudBackend::query (
    ContainerId containerId,
    QueryJoinDelegatePtr queryJoinDelegate )

```

Join multiple tables.

Same as [query\(ContainerId,delegate::QueryDelegatePtr\)](#), but with a [QueryJoinDelegate](#) as delegate, `query↔JoinDelegate`.

See also

[query\(ContainerId,delegate::QueryDelegatePtr\)](#)

Parameters

<i>queryJoinDelegate</i>	Pointer to a QueryJoinDelegate instance, implemented by the user, that receives the response of this query request. I.e., either its callback function onQueryJoinSuccess() or onQueryJoinError() will be called in the event of success or failure respectively.
--------------------------	--

12.8.4.15 queryWithPath() [1/2]

```
cbe::QueryChain cbe::CloudBackend::queryWithPath (
    std::string relativePath,
    cbe::ContainerId queryRoot,
    QueryDelegatePtr delegate )
```

Queries items of a container with a given path.

Note

. . or . relative path options are not permitted.

Only top down search from start point, queryRoot id to downwards path in container tree.

Parameters

<i>relativePath</i>	The relative path from the queryRoot. E.g.: /Documents/Pictures from a queryRoot that has the nested containers Documents and Pictures.
<i>queryRoot</i>	The container id forming the root of this query.
<i>delegate</i>	Pointer to a delegate::QueryDelegate instance that is implemented by the user.

12.8.4.16 queryWithPath() [2/2]

```
cbe::QueryChain cbe::CloudBackend::queryWithPath (
    std::string relativePath,
    QueryDelegatePtr delegate )
```

Queries items of a container with a given path relative the home root container.

Same as [queryWithPath\(std::string, cbe::ContainerId, QueryDelegatePtr\)](#), but with the queryRoot parameter omitted. Instead, here the home root container is used as top container in the search tree.

12.8.4.17 removeListener()

```
void cbe::CloudBackend::removeListener (
    ListenerHandle handle )
```

Delete a specific listener.

Removes a listener that will receive updates as changes occur on the account

Parameters

<i>handle</i>	Handle previously retrieved from method addListener(CloudBackendListenerDelegatePtr) .
---------------	--

12.8.4.18 search() [1/2]

```
cbe::QueryResult cbe::CloudBackend::search (
    cbe::Filter filter,
    cbe::ContainerId containerId,
    QueryDelegatePtr delegate )
```

Select [Object](#)'s with specified key/values using filter.

Pointer to [cbe::delegate::QueryDelegate](#) that is passed into asynchronous version of method search()

Search the whole container with sub-containers related to Objects in the container hierarchy structure.

E.g., Key = Name, Value Contract/Object/Song => Name:Contract1 .

Search handles tags in combination / conjunction of keys and/or key values separated by |.

E.g., Name:*|Country:Sweden|Country:Norway, this would search for objects with key Name of any value and where key Country is either Sweden or Norway.

See [Filter](#).

Parameters

<i>filter</i>	is a cbe::Filter on which you can set how you want data to be ordered when searching. Remember to set the queryString to be keys/tags or key:value pairs that are separated by .
<i>container↔ Id</i>	is the ContainerId of the Container to start the search of Objects in. E.g., if starting in the rootContainer, the whole account will be searched for matching tags, key/value's.
<i>delegate</i>	is the callback pointer to where the API returns from either cache or Server.

12.8.4.19 search() [2/2]

```
cbe::QueryResult cbe::CloudBackend::search (
    std::string tags,
    cbe::ContainerId containerId,
    QueryDelegatePtr delegate )
```

Select [Object](#)'s with specified key/values.

Search the whole container for tags related to [Objects](#) in the container structure.

E.g., Key = Name, Value Contract/Object/Song => Name:Contract1 .

Search handles tags in combination / conjunction of keys and/or key values separated by |.

E.g., Name:*|Country:Sweden|Country:Norway.

This would search for objects with key Name of any value and where key Country is either Sweden or Norway.

Parameters

<i>tags</i>	Is a string of key tags or key:value pairs that are separated by .
<i>container↔ Id</i>	Is the cbe::ContainerId id of the rootContainer to start the search of objects in. E.g., if starting in the rootContainer, the whole account will be searched for matching tags, key/value's.
<i>delegate</i>	Is the callback pointer to where the API returns from either cache or Server.

12.8.4.20 shareManager()

```
cbe::ShareManager cbe::CloudBackend::shareManager ( )
```

Gets the share manager.

Returns

[cbe::ShareManager](#)

12.8.4.21 subscribeManager()

```
cbe::SubscribeManager cbe::CloudBackend::subscribeManager ( )
```

Gets the subscribe manager.

Returns

[cbe::SubscribeManager](#)

12.8.4.22 terminate()

```
void cbe::CloudBackend::terminate ( )
```

Terminates the CloudBackend service.

Shall also be called in connection with a failed log in.

12.8.4.23 version()

```
std::string cbe::CloudBackend::version ( ) const
```

Returns the version number of the SDK.

Returns

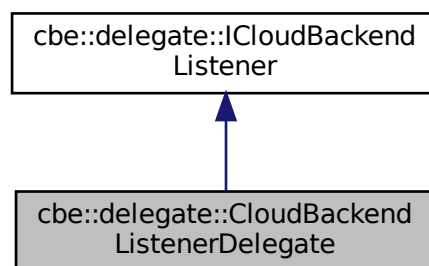
std::string

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

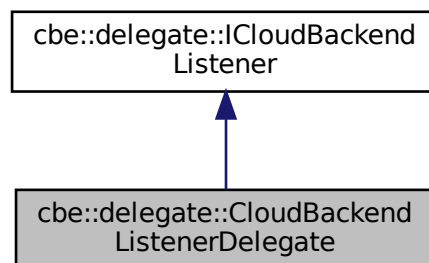
- include/cbe/CloudBackend.h

12.9 cbe::delegate::CloudBackendListenerDelegate Class Reference

Inheritance diagram for cbe::delegate::CloudBackendListenerDelegate:



Collaboration diagram for cbe::delegate::CloudBackendListenerDelegate:



Public Member Functions

- virtual void [onRemoteObjectAdded](#) ([cbe::Object](#) &&object) override
- virtual void [onRemoteObjectMoved](#) ([cbe::Object](#) &&object) override
- virtual void [onRemoteObjectRemoved](#) ([cbe::ItemId](#) objectId, std::string name) override
- virtual void [onRemoteObjectRenamed](#) ([cbe::Object](#) &&object) override
- virtual void [onRemoteContainerAdded](#) ([cbe::Container](#) &&container) override
- virtual void [onRemoteContainerMoved](#) ([cbe::Container](#) &&container) override
- virtual void [onRemoteContainerRemoved](#) ([cbe::ItemId](#) containerId, std::string name) override
- virtual void [onRemoteContainerRenamed](#) ([cbe::Container](#) &&container) override

12.9.1 Member Function Documentation

12.9.1.1 onRemoteContainerAdded()

```
virtual void cbe::delegate::CloudBackendListenerDelegate::onRemoteContainerAdded (
    cbe::Container && container ) [override], [virtual]
```

Called upon successful Create.

Parameters

<i>container</i>	Instance of container that is being created.
------------------	--

Implements [cbe::delegate::ICloudBackendListener](#).

12.9.1.2 onRemoteContainerMoved()

```
virtual void cbe::delegate::CloudBackendListenerDelegate::onRemoteContainerMoved (
    cbe::Container && container ) [override], [virtual]
```

Called upon successful Move.

Parameters

<i>container</i>	Instance of container that is being moved.
------------------	--

Implements [cbe::delegate::ICloudBackendListener](#).

12.9.1.3 onRemoteContainerRemoved()

```
virtual void cbe::delegate::CloudBackendListenerDelegate::onRemoteContainerRemoved (
    cbe::ItemId containerId,
    std::string name ) [override], [virtual]
```

Called upon successful Remove.

Parameters

<i>container↔ Id</i>	Instance of the container that is being Removed.
<i>name</i>	Name of the container that is being removed.

Implements [cbe::delegate::ICloudBackendListener](#).

12.9.1.4 onRemoteContainerRenamed()

```
virtual void cbe::delegate::CloudBackendListenerDelegate::onRemoteContainerRenamed (
    cbe::Container && container ) [override], [virtual]
```

Called upon successful Rename.

Parameters

<i>container</i>	Instance of container that is being Renamed.
------------------	--

Implements [cbe::delegate::ICloudBackendListener](#).

12.9.1.5 onRemoteObjectAdded()

```
virtual void cbe::delegate::CloudBackendListenerDelegate::onRemoteObjectAdded (
    cbe::Object && object ) [override], [virtual]
```

Called upon successful create of an [object](#).

Parameters

<i>object</i>	Instance of object that is being created.
---------------	---

Implements [cbe::delegate::ICloudBackendListener](#).

12.9.1.6 onRemoteObjectMoved()

```
virtual void cbe::delegate::CloudBackendListenerDelegate::onRemoteObjectMoved (
    cbe::Object && object ) [override], [virtual]
```

Called upon successful move of an [object](#).

Parameters

<i>object</i>	Instance of object that is being moved.
---------------	---

Implements [cbe::delegate::ICloudBackendListener](#).

12.9.1.7 onRemoteObjectRemoved()

```
virtual void cbe::delegate::CloudBackendListenerDelegate::onRemoteObjectRemoved (
    cbe::ItemId objectId,
    std::string name ) [override], [virtual]
```

Called upon successful removal of an [object](#).

Parameters

<i>objectId</i>	Instance of the object that is being Removed.
<i>name</i>	Name of the object that is being Removed.

Implements [cbe::delegate::ICloudBackendListener](#).

12.9.1.8 onRemoteObjectRenamed()

```
virtual void cbe::delegate::CloudBackendListenerDelegate::onRemoteObjectRenamed (
```

```
cbe::Object && object ) [override], [virtual]
```

Called upon successful renaming of an [object](#).

Parameters

<i>object</i>	Instance of object that is being renamed.
---------------	---

Implements [cbe::delegate::ICloudBackendListener](#).

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

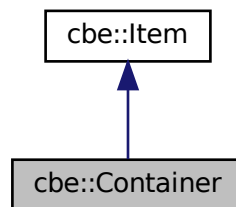
- include/cbe/delegate/CloudBackendListenerDelegate.h

12.10 cbe::Container Class Reference

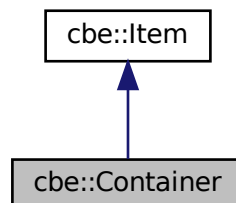
A collection of [Item](#), can also represent a table or folder.

```
#include <Container.h>
```

Inheritance diagram for cbe::Container:



Collaboration diagram for cbe::Container:



Public Types

- using `CreateContainerDelegatePtr` = `delegate::CreateContainerDelegatePtr`
- using `MoveDelegatePtr` = `delegate::container::MoveDelegatePtr`
- using `RenameDelegatePtr` = `delegate::container::RenameDelegatePtr`
- using `RemoveDelegatePtr` = `delegate::container::RemoveDelegatePtr`
- using `CreateObjectDelegatePtr` = `delegate::CreateObjectDelegatePtr`

- using UploadDelegatePtr = delegate::UploadDelegatePtr
- using QueryDelegatePtr = delegate::QueryDelegatePtr
- using QueryJoinDelegatePtr = std::shared_ptr< delegate::QueryJoinDelegate >
- using SetAclDelegatePtr = delegate::AclDelegatePtr
- using GetAclDelegatePtr = delegate::AclDelegatePtr
- using ShareDelegatePtr = delegate::ShareDelegatePtr
- using UnShareDelegatePtr = std::shared_ptr< delegate::UnShareDelegate >
- using PublishDelegatePtr = delegate::PublishDelegatePtr
- using UnPublishDelegatePtr = delegate::UnPublishDelegatePtr
- using UnSubscribeDelegatePtr = delegate::UnSubscribeDelegatePtr

Public Member Functions

- **cbe::Container createContainer** (const std::string &name, CreateContainerDelegatePtr delegate)
Create a container.
- void **move** (cbe::ContainerId dstId, MoveDelegatePtr delegate)
Move a container to a different parent.
- void **rename** (const std::string &name, RenameDelegatePtr delegate)
Change the name of the container.
- void **remove** (RemoveDelegatePtr delegate)
Delete the container.
- **cbe::Object createObject** (std::string name, cbe::KeyValues keyValues, CreateObjectDelegatePtr delegate)
Create a new object.
- **cbe::Object createObject** (std::string name, CreateObjectDelegatePtr delegate)
- **cbe::Object upload** (const std::string &filePath, UploadDelegatePtr delegate)
Upload object to container with file given by filePath.
- **cbe::Object upload** (const std::string &name, const std::string &path, UploadDelegatePtr delegate)
Create an object in current container by uploading a file.
- **cbe::Object upload** (const std::string &name, std::uint64_t length, const char *byteData, UploadDelegatePtr delegate)
Upload from local memory to an object.
- **cbe::QueryChain query** (QueryDelegatePtr queryDelegate)
Select list of objects.
- **cbe::QueryChain query** (Filter filter, QueryDelegatePtr delegate)
Select list of objects using filter.
- **cbe::QueryChainExt query** (delegate::QueryJoinDelegatePtr delegate)
Select list of objects, for join.
- **cbe::QueryChainExt query** (Filter filter, delegate::QueryJoinDelegatePtr delegate)
Select list of objects using filter, for join.
- **cbe::QueryChain queryWithPath** (std::string relativePath, QueryDelegatePtr delegate)
Select list of objects in hierarchy.
- **cbe::QueryResult search** (std::string tags, QueryDelegatePtr delegate)
Search by tags.
- **cbe::QueryResult search** (cbe::Filter filter, QueryDelegatePtr delegate)
Search using filter.
- void **setAcl** (cbe::AclMap aclMap, SetAclDelegatePtr delegate)
set ACL.
- void **getAcl** (GetAclDelegatePtr delegate)
Retrieves its ACL map.
- void **share** (cbe::UserId toUserGroup, std::string description, ShareDelegatePtr delegate)
Make accessible by other user.
- void **unShare** (cbe::ShareId shareId, UnShareDelegatePtr delegate)

Revoke a previous share.

- void `publish` (`cbe::PublishAccess` security, `cbe::PublishVisibility` privacy, `std::string` description, `std::string` password, `PublishDelegatePtr` delegate)

Publishes a container and its content to any user.

- void `unPublish` (`UnPublishDelegatePtr` delegate)

UnPublishes this container.

- void `unSubscribe` (`UnSubscribeDelegatePtr` delegate)

UnSubscribes from this container.

- **Container** (`cbe::DefaultCtor`)

Friends

- class **Account**
- class **CloudBackend**
- class **Database**
- class **Filter**
- class **Group**
- class **QueryChain**

Additional Inherited Members

12.10.1 Detailed Description

A collection of [Item](#), can also represent a table or folder.

12.10.2 Member Typedef Documentation

12.10.2.1 CreateContainerDelegatePtr

using `cbe::Container::CreateContainerDelegatePtr` = `delegate::CreateContainerDelegatePtr`

Pointer to `cbe::delegate::CreateContainerDelegate` that is passed into asynchronous version of method `createContainer()`

12.10.2.2 CreateObjectDelegatePtr

using `cbe::Container::CreateObjectDelegatePtr` = `delegate::CreateObjectDelegatePtr`

Pointer to `cbe::delegate::CreateObjectDelegate` that is passed into asynchronous version of method `createObject()`

12.10.2.3 GetAclDelegatePtr

using `cbe::Container::GetAclDelegatePtr` = `delegate::AclDelegatePtr`

Pointer to `cbe::delegate::AclDelegate` that is passed into asynchronous version of method `getAcl()`

12.10.2.4 MoveDelegatePtr

using `cbe::Container::MoveDelegatePtr` = `delegate::container::MoveDelegatePtr`

Pointer to `cbe::delegate::MoveDelegate` that is passed into asynchronous version of method `move()`

12.10.2.5 PublishDelegatePtr

using `cbe::Container::PublishDelegatePtr` = `delegate::PublishDelegatePtr`

Pointer to `cbe::delegate::PublishDelegate` that is passed into asynchronous version of method `publish()`

12.10.2.6 QueryDelegatePtr

```
using cbe::Container::QueryDelegatePtr = delegate::QueryDelegatePtr
```

Pointer to [cbe::delegate::QueryDelegatePtr](#) that is passed into asynchronous version of method [query\(\)](#)

12.10.2.7 QueryJoinDelegatePtr

```
using cbe::Container::QueryJoinDelegatePtr = std::shared_ptr<delegate::QueryJoinDelegate>
```

Pointer to [cbe::delegate::QueryJoinDelegate](#) that is passed into asynchronous version of method [query\(\)](#)

12.10.2.8 RemoveDelegatePtr

```
using cbe::Container::RemoveDelegatePtr = delegate::container::RemoveDelegatePtr
```

Pointer to [cbe::delegate::RemoveDelegate](#) that is passed into asynchronous version of method [remove\(\)](#)

12.10.2.9 RenameDelegatePtr

```
using cbe::Container::RenameDelegatePtr = delegate::container::RenameDelegatePtr
```

Pointer to [cbe::delegate::container::RenameDelegate](#) that is passed into asynchronous version of method [rename\(\)](#)

12.10.2.10 SetAclDelegatePtr

```
using cbe::Container::SetAclDelegatePtr = delegate::AclDelegatePtr
```

Pointer to [cbe::delegate::SearchDelegate](#) that is passed into asynchronous version of method [search\(\)](#) Pointer to [cbe::delegate::AclDelegate](#) that is passed into asynchronous version of method [setAcl\(\)](#)

12.10.2.11 ShareDelegatePtr

```
using cbe::Container::ShareDelegatePtr = delegate::ShareDelegatePtr
```

Pointer to [cbe::delegate::ShareDelegate](#) that is passed into asynchronous version of method [share\(\)](#)

12.10.2.12 UnPublishDelegatePtr

```
using cbe::Container::UnPublishDelegatePtr = delegate::UnPublishDelegatePtr
```

Pointer to [cbe::delegate::UnPublishDelegatePtr](#) that is passed into asynchronous version of method [unPublish\(\)](#)

12.10.2.13 UnShareDelegatePtr

```
using cbe::Container::UnShareDelegatePtr = std::shared_ptr<delegate::UnShareDelegate>
```

Pointer to [cbe::delegate::UnShareDelegate](#) that is passed into asynchronous version of method [unShare\(\)](#)

12.10.2.14 UnSubscribeDelegatePtr

```
using cbe::Container::UnSubscribeDelegatePtr = delegate::UnSubscribeDelegatePtr
```

Pointer to [cbe::delegate::UnPublishDelegate](#) that is passed into asynchronous version of method [unPublish\(\)](#) Pointer to [cbe::delegate::UnSubscribeDelegate](#) that is passed into asynchronous version of method [unSubscribe\(\)](#)

12.10.2.15 UploadDelegatePtr

```
using cbe::Container::UploadDelegatePtr = delegate::UploadDelegatePtr
```

Pointer to [cbe::delegate::UploadDelegate](#) that is passed into asynchronous version of methods:

- [upload\(const std::string&, UploadDelegatePtr\)](#)
- [upload\(const std::string&, const std::string&, UploadDelegatePtr\)](#)
- [upload\(const std::string&, std::uint64_t, const char*, UploadDelegatePtr\)](#)

12.10.3 Member Function Documentation

12.10.3.1 createContainer()

```
cbe::Container cbe::Container::createContainer (
    const std::string & name,
    CreateContainerDelegatePtr delegate )
```

Create a container.

Creates a container inside this container to be used for adding objects.

Parameters

<i>name</i>	Name of the container to be created.
<i>delegate</i>	Pointer to a delegate::CreateContainerDelegate instance that is implemented by the user.

12.10.3.2 createObject() [1/2]

```
cbe::Object cbe::Container::createObject (
    std::string name,
    cbe::KeyValues keyValues,
    CreateObjectDelegatePtr delegate )
```

Create a new object.

Creates an object with indexed tags or indexed tags + non indexed tags a.k.a. metadata, key/value pairs.

Parameters

<i>name</i>	Name of the object.
<i>keyValues</i>	Optional map of key/value pairs (metadata).
<i>delegate</i>	Pointer to a delegate::CreateObjectDelegate instance that is implemented by the user.

Note

If an old object with the same name already exists that will be removed.

The object name may not contain characters < & : /

Any key name must start with a letter or _

The following key names are reserved and should not be used: category, content, id, link and date

Key names are case sensitive, hence variations with uppercase are permitted.

12.10.3.3 createObject() [2/2]

```
cbe::Object cbe::Container::createObject (
    std::string name,
    CreateObjectDelegatePtr delegate )
```

Same as [createObject\(std::string, cbe::KeyValues, CreateObjectDelegatePtr\)](#), but without the `keyValues` parameter.

12.10.3.4 getAcl()

```
void cbe::Container::getAcl (
    GetAclDelegatePtr delegate )
```

Retrieves its ACL map.

get the Access Control List ACL of the [Container](#).

Parameters

<i>delegate</i>	Pointer to a delegate::AclDelegate instance that is implemented by the user.
-----------------	--

12.10.3.5 move()

```
void cbe::Container::move (
    cbe::ContainerId dstId,
    MoveDelegatePtr delegate )
```

Move a container to a different parent.

Used to move container with its content to user specified location e.g., other container or to root container.

Parameters

<i>dstId</i>	id of the container to which it should be moved to.
<i>delegate</i>	Pointer to a delegate::container::MoveDelegate instance that is implemented by the user.

12.10.3.6 publish()

```
void cbe::Container::publish (
    cbe::PublishAccess security,
    cbe::PublishVisibility privacy,
    std::string description,
    std::string password,
    PublishDelegatePtr delegate )
```

Publishes a container and its content to any user.

Asynchronous version of this service function.

Can be revoked with [unPublish\(\)](#)

Parameters

<i>security</i>	A cbe::PublishAccess enum
<i>privacy</i>	A cbe::PublishVisibility enum
<i>description</i>	Free text
<i>password</i>	Password
<i>delegate</i>	Pointer to a delegate::PublishDelegate instance that is implemented by the user.

12.10.3.7 query() [1/4]

```
cbe::QueryChainExt cbe::Container::query (
    delegate::QueryJoinDelegatePtr delegate )
```

Select list of objects, for join.

In line with function [CloudBackend::query\(ContainerId, delegate::QueryJoinDelegatePtr\)](#), but with its `ContainerId` parameter excluded.

See also

[CloudBackend::query\(ContainerId, delegate::QueryJoinDelegatePtr\)](#)

12.10.3.8 query() [2/4]

```
cbe::QueryChainExt cbe::Container::query (
    Filter filter,
    delegate::QueryJoinDelegatePtr delegate )
```

Select list of objects using filter, for join.

In line with function [CloudBackend::query\(ContainerId, Filter, delegate::QueryJoinDelegatePtr\)](#), but with its `ContainerId` parameter excluded.

See also

[CloudBackend::query\(ContainerId, Filter, delegate::QueryJoinDelegatePtr\)](#)

12.10.3.9 query() [3/4]

```
cbe::QueryChain cbe::Container::query (
    Filter filter,
    QueryDelegatePtr delegate )
```

Select list of objects using filter.

In line with function [CloudBackend::query\(ContainerId, Filter, QueryDelegatePtr\)](#), but with its `ContainerId` parameter excluded.

See also

[CloudBackend::query\(ContainerId, Filter, QueryDelegatePtr\)](#)

12.10.3.10 query() [4/4]

```
cbe::QueryChain cbe::Container::query (
    QueryDelegatePtr queryDelegate )
```

Select list of objects.

In line with function [CloudBackend::query\(ContainerId, QueryDelegatePtr\)](#), but with its `ContainerId` parameter excluded.

See also

[CloudBackend::query\(ContainerId, QueryDelegatePtr\)](#)

12.10.3.11 queryWithPath()

```
cbe::QueryChain cbe::Container::queryWithPath (
    std::string relativePath,
    QueryDelegatePtr delegate )
```

Select list of objects in hierarchy.

Pointer to `cbe::delegate::queryWithPathDelegate` that is passed into asynchronous version of method [queryWithPath\(\)](#)

Queries the container with a given relative path, returns container with objects.

E.g. `/Documents/Pictures` will return objects and subContainers for Pictures.

Note

`..` or `.` path options are not available, top down Paths in the container tree are available.

Parameters

<i>relativePath</i>	container path, e.g. <code>/Documents/Pictures</code>
<i>delegate</i>	Pointer to a delegate::QueryDelegate instance that is implemented by the user.

12.10.3.12 remove()

```
void cbe::Container::remove (
    RemoveDelegatePtr delegate )
```

Delete the container.

Removes/deletes the container and all its content.

Parameters

<i>delegate</i>	Pointer to a delegate::container::RemoveDelegate instance that is implemented by the user.
-----------------	--

12.10.3.13 rename()

```
void cbe::Container::rename (
    const std::string & name,
    RenameDelegatePtr delegate )
```

Change the name of the container.

Rename the container.

Parameters

<i>name</i>	New name of the container.
<i>delegate</i>	Pointer to a delegate::container::RenameDelegate instance that is implemented by the user.

12.10.3.14 search() [1/2]

```
cbe::QueryResult cbe::Container::search (
    cbe::Filter filter,
    QueryDelegatePtr delegate )
```

Search using filter.

Search the whole container with sub-containers related to Objects in the container hierarchy structure.

E.g. Key = Name, Value Contract/Object/Song => Name:Contract1.

Search handles tags in combination / conjunction of keys and/or key values separated by |.

E.g. Name:*|Country:Sweden|Country:Norway, this would search for objects with key Name of any value and where key Country is either Sweden or Norway.

Parameters

<i>filter</i>	is a cbe::Filter on which you can set how you want data to be ordered when searching. Remember to set the queryString to be keys/tags or key:value pairs that are separated by .
<i>delegate</i>	Pointer to a delegate::QueryDelegate instance that is implemented by the user.

12.10.3.15 search() [2/2]

```
cbe::QueryResult cbe::Container::search (
    std::string tags,
    QueryDelegatePtr delegate )
```

Search by tags.

Pointer to [cbe::delegate::QueryDelegate](#) that is passed into asynchronous version of method [search\(\)](#)

Search the whole container for tags related to Objects in the container structure.

E.g. Key = Name, Value Contract/Object/Song => Name:Contract1.

Search handles tags in combination of conjunctions of keys and/or key values separated by |.

E.g. Name:*|Country:Sweden|Country:Norway, this would search for objects with key Name of any value and where key Country is either Sweden or Norway.

Parameters

<i>tags</i>	is a string of key tags or key:value pairs that are separated by .
<i>delegate</i>	Pointer to a delegate::QueryDelegate instance that is implemented by the user.

12.10.3.16 setAcl()

```
void cbe::Container::setAcl (
    cbe::AclMap aclMap,
    SetAclDelegatePtr delegate )
```

set ACL.

Set the Access Control List ACL for the container. For containers set does set the whole container tree, so all its sub items as well. Remember this is set and not update so every time you set all ids that should be there should be added.

Parameters

<i>aclMap</i>	The desired permissions for current container.
<i>delegate</i>	Pointer to a delegate::AclDelegate instance that is implemented by the user.

12.10.3.17 share()

```
void cbe::Container::share (
    cbe::UserId toUserGroup,
    std::string description,
    ShareDelegatePtr delegate )
```

Make accessible by other user.

Shares a container and its content to a user. This provides the user the ability to access what has been shared to them via the listAvailableShares command. To allow users to view and change shared information see ACL .

Note

At present Sharing the container gives the user read permissions for the container and all its sub-items, this might change in the future.

Parameters

<i>toUserGroup</i>	takes a user id or group id to share to.
<i>description</i>	names the specific share between you and the user/group.
<i>delegate</i>	Pointer to a delegate::ShareDelegate instance that is implemented by the user.

12.10.3.18 unPublish()

```
void cbe::Container::unPublish (
    UnPublishDelegatePtr delegate )
```

UnPublishes this container.

Asynchronous version of this service function.

Revokes previous [publish\(\)](#).

Parameters

<i>delegate</i>	Gets notified when the container has been unPublished (or if there was an error)
-----------------	--

12.10.3.19 unShare()

```
void cbe::Container::unShare (
    cbe::ShareId shareId,
    UnShareDelegatePtr delegate )
```

Revoke a previous share.

unShare the container to a specific shareId created when sharing. Each share is unique between user/group and the one sharing. This is represented with a unique share id.

Parameters

<i>shareId</i>	is as mentioned the unique id for a share between the owner and other user/group.
<i>delegate</i>	Pointer to a delegate::UnShareDelegate instance that is implemented by the user.

12.10.3.20 unSubscribe()

```
void cbe::Container::unSubscribe (
    UnSubscribeDelegatePtr delegate )
```

UnSubscribes from this container.

Asynchronous version of this service function.

Revokes the subscription previously established with [cbe::SubscribeManager::subscribe\(\)](#)

Parameters

<i>delegate</i>	Gets notified when the container has been unSubscribed (or if there was an error)
-----------------	---

12.10.3.21 upload() [1/3]

```
cbe::Object cbe::Container::upload (
    const std::string & filePath,
    UploadDelegatePtr delegate )
```

Upload object to container with file given by filePath.

See [upload\(const std::string&,const std::string& path,UploadDelegatePtr\)](#)

Parameters

<i>filePath</i>	Fully qualified file name. I.e., the path, relative or absolute, including file name.
-----------------	---

Example

Async creation of a [cbe::Object](#) by using **Container::upload()**

```
#include "cbe/Object.h"
#include "cbe/Container.h"
#include "cbe/delegate/UploadDelegate.h"
#include <condition_variable>
#include <fstream>           // std::ofstream
#include <mutex>
~~~
struct MyUploadDelegate : cbe::delegate::UploadDelegate {
    std::mutex            mutex{};
```

```

std::condition_variable conditionVariable{};
// Indicates operation completed - success or failure
bool called{};
// Default construct the result object. If the method
// onUploadSuccess() has not been called, this default constructed state
// implies that the object is not valid
cbe::Object object{ cbe::DefaultCtor{} };
void onUploadSuccess(cbe::Object&& object) final {
    {
        std::lock_guard<std::mutex> lock{mutex};
        // Change state of object member to indicate success
        this->object = std::move(object);
        // Indicate operation completed, member object indicates success
        called = true;
    }
    conditionVariable.notify_one();
}
void onUploadError(cbe::delegate::TransferError&& transferError,
                  cbe::util::Context&& context) final {
    {
        std::lock_guard<std::mutex> lock{mutex};
        // Put object into the default constructed state to indicate no-success
        object = cbe::Object{ cbe::DefaultCtor{} };
        // Indicate operation completed, member object indicates failure
        called = true;
    }
    conditionVariable.notify_one();
}
cbe::Object waitForRsp() {
    std::unique_lock<std::mutex> lock{mutex};
    conditionVariable.wait(lock, [this]{ return called; });
    // Reset called flag, so current delegate instance can be reused
    called = false;
    // If object is still in its default constructed state, this implies a
    // failed cbe::object::upload() operation
    return object;
}
}; // struct MyUploadDelegate
~~~
constexpr const char* const uploadPath = "/tmp/upload/";
constexpr const char* const myObjectFileName = "myObject";
const std::string qualFileName =
    std::string{uploadPath} + myObjectFileName;
std::ofstream ofs{qualFileName};
ofs << "Line 11\n" << "Line 12\n" << "Line 13\n" << std::flush;
ofs.close();
// Access container previously created with cbe::Container::createContainer()
cbe::Container myContainer = ~~~;
std::shared_ptr<MyUploadDelegate> myUploadDelegate =
    std::make_shared<MyUploadDelegate>();
// Create an object named after file name in variable qualFileName
myContainer.upload(qualFileName, myUploadDelegate);
// Continuously, use the Object instance passed into the delegate.
cbe::Object object = myUploadDelegate->waitForRsp();
// Check if upload was successful
if (!object) {
    // Not, bail out
    ~~~
}
}
~~~

```

Continues at: [Async retrieving the Stream attached to a cbe::Object with the Object::getStreams\(\)](#)"

12.10.3.22 upload() [2/3]

```

cbe::Object cbe::Container::upload (
    const std::string & name,
    const std::string & path,
    UploadDelegatePtr delegate )

```

Create an object in current container by uploading a file.

Object is named by name, residing at path.

The **object** being created is instantly returned with a temporary id. When the response is retrieved from the server, via callback method [onUploadSuccess\(\)](#) the object will be updated with its final unique id.

Parameters

<i>name</i>	Name of local file name. The object, that is created, will be given the same name.
-------------	--

Parameters

<i>path</i>	Path to local folder where the file is located. The can be relative or absolute and must end with a slash (/) .
<i>delegate</i>	Pointer to a delegate::UploadDelegate instance that is implemented by the user.

Returns

The created [Object](#), first with its temporary id, and after successful with is final id.

12.10.3.23 upload() [3/3]

```
cbe::Object cbe::Container::upload (
    const std::string & name,
    std::uint64_t length,
    const char * byteData,
    UploadDelegatePtr delegate )
```

Upload from local memory to an object.

Parameters

<i>name</i>	name that the uploaded object will get
<i>length</i>	size of file in Bytes
<i>byteData</i>	(char pointer to an array containing the data).
<i>delegate</i>	Pointer to a delegate::UploadDelegate instance that is implemented by the user.

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

- include/cbe/Container.h

12.11 cbe::util::Context Struct Reference

Public Types

- using **ReportFn** = std::function< void(std::ostream &);>

Public Member Functions

- **Context** (ReportFn &&reportFn, const char fnName[])
- std::string **report** () const

Public Attributes

- std::string **fnName** {}

Friends

- std::ostream & **operator**<< (std::ostream &os, const [Context](#) &context)

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

- include/cbe/util/Context.h

12.12 cbe::delegate::CreateAccountDelegate Class Reference

```
#include <CreateAccountDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [UserId](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onCreateAccountSuccess](#) ([UserId](#) &&userId)=0
- virtual void [onCreateAccountError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.12.1 Detailed Description

Delegate class for the asynchronous version of method:

- [CloudBackend::CreateAccount](#)

12.12.2 Member Function Documentation

12.12.2.1 [onCreateAccountError\(\)](#)

```
virtual void cbe::delegate::CreateAccountDelegate::onCreateAccountError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called upon failed log in.

Parameters

<i>error</i>	Error information passed from CloudBackend SDK.
<i>context</i>	Additional context information about the original service call that has failed.

12.12.2.2 [onCreateAccountSuccess\(\)](#)

```
virtual void cbe::delegate::CreateAccountDelegate::onCreateAccountSuccess (
    UserId && userId ) [pure virtual]
```

Called upon successful account creation.

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

- [include/cbe/delegate/CreateAccountDelegate.h](#)

12.13 [cbe::delegate::CreateContainerDelegate](#) Class Reference

```
#include <CreateContainerDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Container](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onCreateContainerSuccess](#) ([cbe::Container](#) &&container)=0
- virtual void [onCreateContainerError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.13.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Container::createContainer\(\)](#)

12.13.2 Member Function Documentation

12.13.2.1 onCreateContainerError()

```
virtual void cbe::delegate::CreateContainerDelegate::onCreateContainerError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.13.2.2 onCreateContainerSuccess()

```
virtual void cbe::delegate::CreateContainerDelegate::onCreateContainerSuccess (
    cbe::Container && container ) [pure virtual]
```

Called upon successful CreateContainer.

Parameters

<i>container</i>	Instance of Container that is being created.
------------------	--

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

- include/cbe/delegate/CreateContainerDelegate.h

12.14 cbe::delegate::CreateGroupDelegate Class Reference

```
#include <CreateGroupDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Group](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onCreateGroupSuccess](#) ([cbe::Group](#) &&group)=0
- virtual void [onCreateGroupError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.14.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Group::createGroup](#)

12.14.2 Member Function Documentation

12.14.2.1 onCreateGroupError()

```
virtual void cbe::delegate::CreateGroupDelegate::onCreateGroupError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.14.2.2 onCreateGroupSuccess()

```
virtual void cbe::delegate::CreateGroupDelegate::onCreateGroupSuccess (
    cbe::Group && group ) [pure virtual]
```

Called upon successful create group.

Parameters

<i>group</i>	Instance of a cbe::Group .
--------------	--

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

- include/cbe/delegate/CreateGroupDelegate.h

12.15 cbe::delegate::CreateObjectDelegate Class Reference

```
#include <CreateObjectDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Object](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onCreateObjectSuccess](#) ([cbe::Object](#) &&object)=0
- virtual void [onCreateObjectError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.15.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Container::createObject](#)

12.15.2 Member Function Documentation

12.15.2.1 onCreateObjectError()

```
virtual void cbe::delegate::CreateObjectDelegate::onCreateObjectError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.15.2.2 onCreateObjectSuccess()

```
virtual void cbe::delegate::CreateObjectDelegate::onCreateObjectSuccess (
    cbe::Object && object ) [pure virtual]
```

Called upon successful CreateObject.

Parameters

<i>object</i>	Instance of Object that is being created.
---------------	---

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

- include/cbe/delegate/CreateObjectDelegate.h

12.16 cbe::delegate::CreateRoleDelegate Class Reference

```
#include <CreateRoleDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Role](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onCreateRoleSuccess](#) ([cbe::Role](#) &&role)=0
- virtual void [onCreateRoleError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.16.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Role::createRole](#)

12.16.2 Member Function Documentation

12.16.2.1 onCreateRoleError()

```
virtual void cbe::delegate::CreateRoleDelegate::onCreateRoleError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.16.2.2 onCreateRoleSuccess()

```
virtual void cbe::delegate::CreateRoleDelegate::onCreateRoleSuccess (
    cbe::Role && role ) [pure virtual]
```

Called upon successful create [Role](#).

Parameters

<i>Role</i>	the role id of the created role.
-----------------------------	----------------------------------

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

- include/cbe/delegate/CreateRoleDelegate.h

12.17 cbe::Database Class Reference

A database.

```
#include <Database.h>
```

Public Member Functions

- `std::string name () const`
The database name.
- `cbe::DatabaseId databaseld () const`
The databaseld number.
- `cbe::ContainerId containerId () const`
The containerId number.
- `cbe::UserId ownerId () const`
The userId number of the owner.
- `cbe::Date created () const`
Timestamp of when it was created.
- `bool readLocked () const`
Checks if it has a read lock.
- `bool writeLocked () const`
Checks if it has a write lock.
- `cbe::Container rootContainer () const`
The top container of the database.
- **Database** (`cbe::DefaultCtor`)
- **operator bool** () const

12.17.1 Detailed Description

A database.

Example

Get the tenant `Container` object

```
cbe::DataBases myDatabases = cloudBackend.account().databases();
cbe::Container tenantContainer{ cbe::DefaultCtor{} };
for (auto myEntry : myDatabases) {
    if ( myEntry.first == "tenant" ) {
        tenantContainer = myEntry.second.rootContainer();
    }
}
return tenantContainer;
```

12.17.2 Member Function Documentation

12.17.2.1 containerId()

```
cbe::ContainerId cbe::Database::containerId ( ) const
```

The containerId number.

Refers to the top rootContainer of the database.

Returns

`cbe::ContainerId`

12.17.2.2 created()

```
cbe::Date cbe::Database::created ( ) const
```

Timestamp of when it was created.

Given in unix epoch number format.

Returns

[cbe::Date](#)

12.17.2.3 databaseId()

```
cbe::DatabaseId cbe::Database::databaseId ( ) const
```

The databaseId number.

Returns

[cbe::DatabaseId](#)

12.17.2.4 name()

```
std::string cbe::Database::name ( ) const
```

The database name.

Returns

[std::string](#)

12.17.2.5 ownerId()

```
cbe::UserId cbe::Database::ownerId ( ) const
```

The userId number of the owner.

Returns

[cbe::UserId](#)

12.17.2.6 readLocked()

```
bool cbe::Database::readLocked ( ) const
```

Checks if it has a read lock.

Returns

[true](#)

[false](#)

12.17.2.7 rootContainer()

```
cbe::Container cbe::Database::rootContainer ( ) const
```

The top container of the database.

The whole container object.

Returns

[cbe::Container](#)

12.17.2.8 writeLocked()

`bool cbe::Database::writeLocked ( ) const`
Checks if it has a write lock.

Returns

true
false

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

- include/cbe/Database.h

12.18 cbe::delegate::DownloadBinaryDelegate Class Reference

```
#include <DownloadBinaryDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [DownloadBinarySuccess](#)
- using **Error** = [TransferError](#)

Public Member Functions

- virtual void [onDownloadBinarySuccess](#) ([cbe::Object](#) &&object, std::unique_ptr< char[]> data)=0
- virtual void [onDownloadBinaryError](#) ([cbe::delegate::TransferError](#) &&transferError, [cbe::util::Context](#) &&context)=0
- virtual void [onChunkReceived](#) ([cbe::Object](#) &&object, std::uint64_t received, std::uint64_t total)

12.18.1 Detailed Description

Delegate class for the asynchronous version of method:

- `cbe::Object::download(DownloadBinaryDelegatePtr)`

12.18.2 Member Function Documentation

12.18.2.1 onChunkReceived()

```
virtual void cbe::delegate::DownloadBinaryDelegate::onChunkReceived (
    cbe::Object && object,
    std::uint64_t received,
    std::uint64_t total ) [virtual]
```

Called when a chunk of data has been received.

Parameters

<i>object</i>	Object associated with current download.
<i>received</i>	Size, in number of bytes, of the received chunk.
<i>total</i>	Total number of bytes received so far.

12.18.2.2 onDownloadBinaryError()

```
virtual void cbe::delegate::DownloadBinaryDelegate::onDownloadBinaryError (
    cbe::delegate::TransferError && transferError,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.18.2.3 onDownloadBinarySuccess()

```
virtual void cbe::delegate::DownloadBinaryDelegate::onDownloadBinarySuccess (
    cbe::Object && object,
    std::unique_ptr< char[] > data ) [pure virtual]
```

Called upon successful Download.

Parameters

<i>object</i>	Instance of cbe::Object that is being downloaded.
<i>data</i>	The downloaded data of the object. The provided memory is an array of bytes (<code>char</code>) dynamically allocated by the SDK, and managed by the <code>std::unique_ptr</code> . So, no further precautions needed concerning the the disposal of this memory.

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

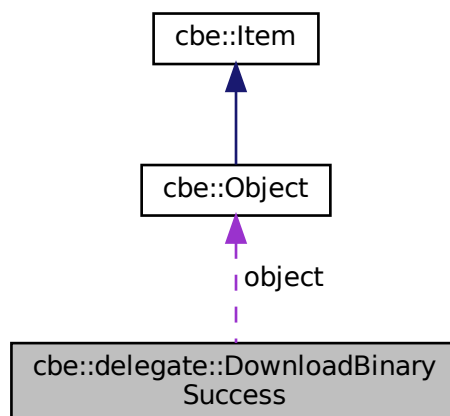
- `include/cbe/delegate/DownloadBinaryDelegate.h`

12.19 cbe::delegate::DownloadBinarySuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::DownloadBinaryDelegate::onDownloadBinarySuccess](#)

```
#include <DownloadBinarySuccess.h>
```

Collaboration diagram for `cbe::delegate::DownloadBinarySuccess`:



Public Member Functions

- **DownloadBinarySuccess** ([cbe::DefaultCtor](#))
- **DownloadBinarySuccess** ([cbe::Object](#) &&object, `std::unique_ptr< char[] > data`)

- `operator bool () const`

Checks if current instance is valid.

Public Attributes

- `cbe::Object object {cbe::DefaultCtor{}}`
- `std::unique_ptr< char[]> data {}`

12.19.1 Detailed Description

Convenience type that bundles all parameters passed to method `cbe::delegate::DownloadBinaryDelegate::onDownloadBinarySuccess`

12.19.2 Member Function Documentation

12.19.2.1 `operator bool()`

```
cbe::delegate::DownloadBinarySuccess::operator bool ( ) const [explicit]
```

Checks if current instance is valid.

Returns

`true`: is valid

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

- `include/cbe/delegate/DownloadBinarySuccess.h`

12.20 `cbe::delegate::DownloadDelegate` Class Reference

```
#include <DownloadDelegate.h>
```

Classes

- struct `ErrorInfo`

Public Types

- using `Success` = `DownloadSuccess`
- using `Error` = `cbe::delegate::TransferError`

Public Member Functions

- virtual void `onDownloadSuccess` (`cbe::Object` &&object, `std::string` path)=0
- virtual void `onDownloadError` (`cbe::delegate::TransferError` &&transferError, `cbe::util::Context` &&context)=0
- virtual void `onChunkReceived` (`cbe::Object` &&object, `std::uint64_t` received, `std::uint64_t` total)

12.20.1 Detailed Description

Delegate class for the asynchronous version of method:

- `cbe::Object::download(const std::string&, DownloadDelegatePtr)`
- `cbe::Object::downloadStream(const std::string&,cbe::Stream,DownloadDelegatePtr)`

12.20.2 Member Function Documentation

12.20.2.1 onChunkReceived()

```
virtual void cbe::delegate::DownloadDelegate::onChunkReceived (
    cbe::Object && object,
    std::uint64_t received,
    std::uint64_t total ) [virtual]
```

Called when a chunk of data has been received.

Parameters

<i>object</i>	Object associated with current download.
<i>received</i>	Size, in number of bytes, of the received chunk.
<i>total</i>	Total number of bytes received so far.

12.20.2.2 onDownloadError()

```
virtual void cbe::delegate::DownloadDelegate::onDownloadError (
    cbe::delegate::TransferError && transferError,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.20.2.3 onDownloadSuccess()

```
virtual void cbe::delegate::DownloadDelegate::onDownloadSuccess (
    cbe::Object && object,
    std::string path ) [pure virtual]
```

Called upon successful Download.

Parameters

<i>object</i>	Instance of object that is being Downloaded.
<i>path</i>	Path of object to be downloaded.

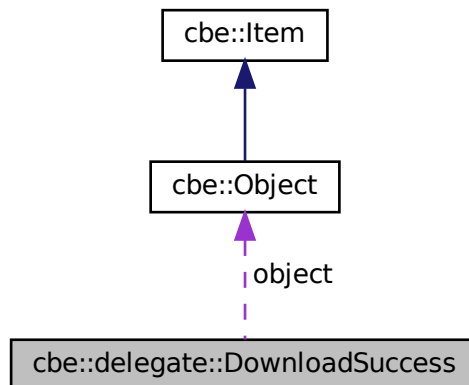
The documentation for this class was generated from the following file:

- include/cbe/delegate/DownloadDelegate.h

12.21 cbe::delegate::DownloadSuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::DownloadDelegate::onDownloadSuccess](#).
`#include <DownloadSuccess.h>`

Collaboration diagram for `cbe::delegate::DownloadSuccess`:



Public Member Functions

- **DownloadSuccess** ([cbe::DefaultCtor](#))
- **DownloadSuccess** ([cbe::Object](#) &&object, std::string path)
- **operator bool** () const

Checks if current instance is valid.

Public Attributes

- [cbe::Object](#) **object** {[cbe::DefaultCtor](#){}}
- std::string **path** {}

12.21.1 Detailed Description

Convenience type that bundles all parameters passed to method [cbe::delegate::DownloadDelegate::onDownloadSuccess](#).

12.21.2 Member Function Documentation

12.21.2.1 operator bool()

```
cbe::delegate::DownloadSuccess::operator bool ( ) const [explicit]
```

Checks if current instance is valid.

Returns

`true`: is valid

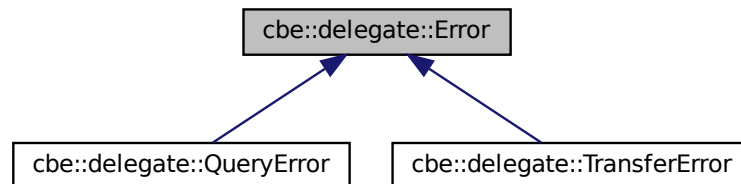
The documentation for this class was generated from the following file:

- include/cbe/delegate/DownloadSuccess.h

12.22 cbe::delegate::Error Class Reference

```
#include <Error.h>
```

Inheritance diagram for cbe::delegate::Error:



Public Member Functions

- `operator bool () const`

Public Attributes

- `ErrorCode errorCode {}`
- `std::string reason {}`
- `std::string message {}`

Friends

- `std::ostream & operator<< (std::ostream &os, const Error &error)`

12.22.1 Detailed Description

Entity class containing the error information

12.22.2 Member Function Documentation

12.22.2.1 operator bool()

```
cbe::delegate::Error::operator bool ( ) const [explicit]
```

Checks whether current object holds error information.

Returns

`true` implies that current object contains an error, while `false` indicates no error.

12.22.3 Friends And Related Function Documentation

12.22.3.1 operator<<

```
std::ostream& operator<< (
    std::ostream & os,
    const Error & error ) [friend]
```

Output stream operator for CloudBackend::LoginDelegate::Error.

12.22.4 Member Data Documentation

12.22.4.1 errorCode

`ErrorCode` `cbe::delegate::Error::errorCode` {}

Mimics the general error code encoding in the www.

See also

[Wikipedia: List of HTTP status codes](#)

12.22.4.2 message

`std::string` `cbe::delegate::Error::message` {}

Human readable additional information about the error.

12.22.4.3 reason

`std::string` `cbe::delegate::Error::reason` {}

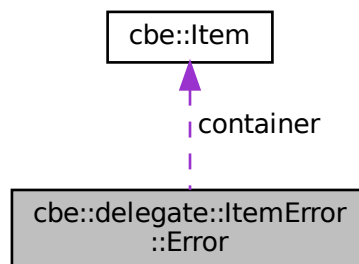
Human readable description of the reason for the failure.

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

- `include/cbe/delegate/Error.h`

12.23 cbe::delegate::ItemError::Error Struct Reference

Collaboration diagram for `cbe::delegate::ItemError::Error`:



Public Member Functions

- **Error** (`Item` container, `ItemType` type, `ErrorCode` errorCode, `std::string` &&reason, `std::string` &&message)

Public Attributes

- `Item` container {`cbe::DefaultCtor`{}}
- `ItemType` type {}
- `ErrorCode` errorCode {}
- `std::string` reason {}
- `std::string` message {}

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

- include/cbe/delegate/ItemDelegateErrors.h

12.24 cbe::delegate::JoinError::Error Class Reference

Public Member Functions

- **Error** ([ErrorCode](#) errorCode, std::string &&reason, std::string &&message)

Public Attributes

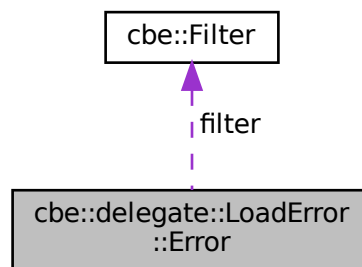
- [ErrorCode](#) **errorCode** {}
- std::string **reason** {}
- std::string **message** {}

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

- include/cbe/delegate/ItemDelegateErrors.h

12.25 cbe::delegate::LoadError::Error Class Reference

Collaboration diagram for cbe::delegate::LoadError::Error:



Public Member Functions

- **Error** ([Filter](#) &&filter, [ErrorCode](#) errorCode, std::string &&reason, std::string &&message)

Public Attributes

- [Filter](#) **filter** {}
- [ErrorCode](#) **errorCode** {}
- std::string **reason** {}
- std::string **message** {}

12.25.1 Member Data Documentation

12.25.1.1 errorCode

```
ErrorCode cbe::delegate::LoadError::Error::errorCode {}
```

Mimics the general error code encoding in the www.

See also

[Wikipedia: List of HTTP status codes](#)

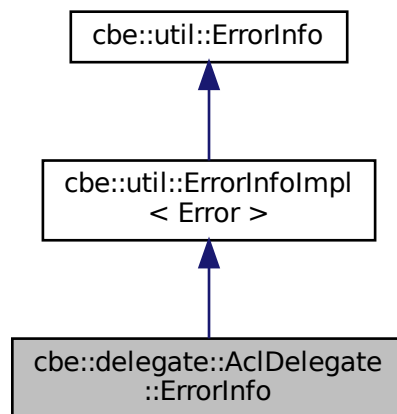
The documentation for this class was generated from the following file:

- include/cbe/delegate/ItemDelegateErrors.h

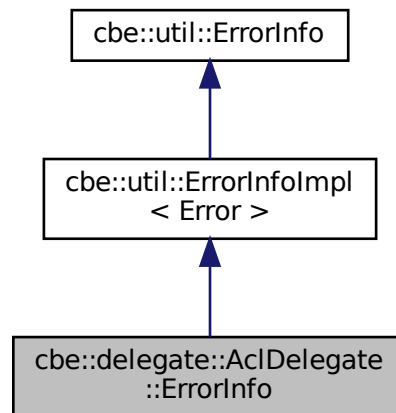
12.26 cbe::delegate::AclDelegate::ErrorInfo Struct Reference

```
#include <AclDelegate.h>
```

Inheritance diagram for cbe::delegate::AclDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::AcIDelegate::ErrorInfo:



Additional Inherited Members

12.26.1 Detailed Description

Contains all information about a failed GetACL.

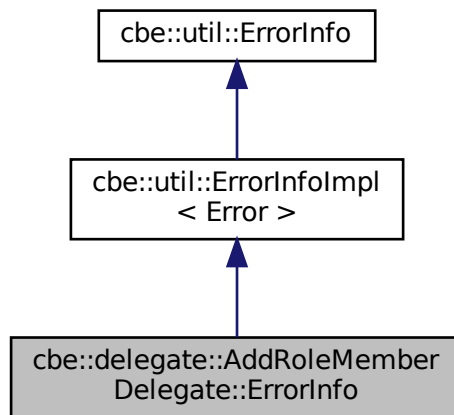
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/AcIDelegate.h`

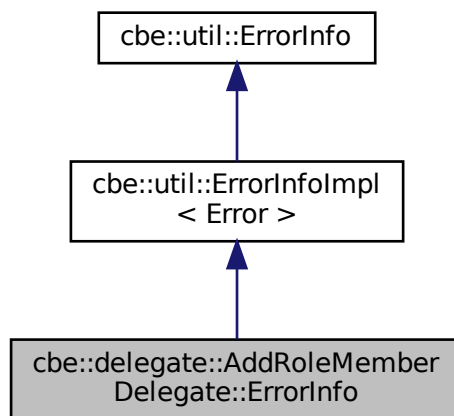
12.27 cbe::delegate::AddRoleMemberDelegate::ErrorInfo Struct Reference

```
#include <AddRoleMemberDelegate.h>
```

Inheritance diagram for `cbe::delegate::AddRoleMemberDelegate::ErrorInfo`:



Collaboration diagram for `cbe::delegate::AddRoleMemberDelegate::ErrorInfo`:



Additional Inherited Members

12.27.1 Detailed Description

Contains all information about a failed `AddMember`.

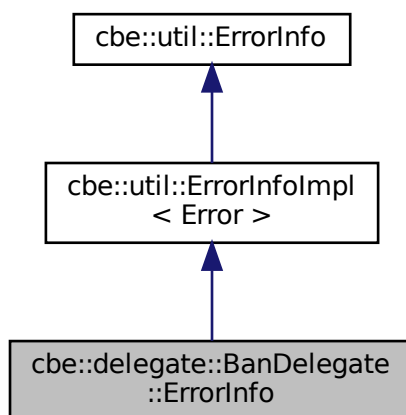
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/AddRoleMemberDelegate.h`

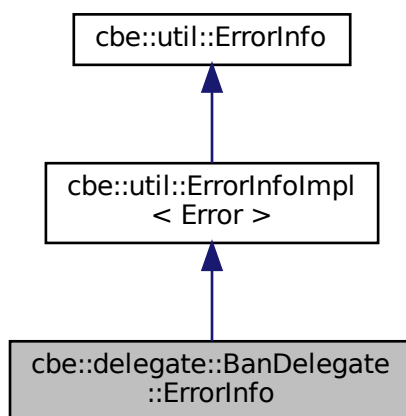
12.28 `cbe::delegate::BanDelegate::ErrorInfo` Struct Reference

```
#include <BanDelegate.h>
```

Inheritance diagram for cbe::delegate::BanDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::BanDelegate::ErrorInfo:



Additional Inherited Members

12.28.1 Detailed Description

Contains all information about a failed Ban.

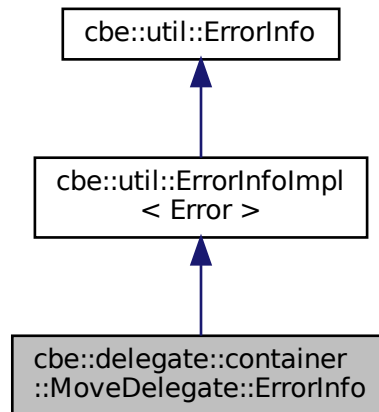
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/BanDelegate.h`

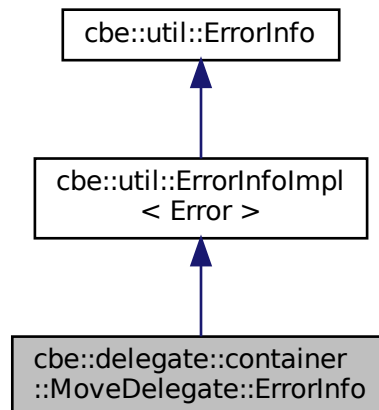
12.29 cbe::delegate::container::MoveDelegate::ErrorInfo Struct Reference

```
#include <MoveDelegate.h>
```

Inheritance diagram for cbe::delegate::container::MoveDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::container::MoveDelegate::ErrorInfo:



Additional Inherited Members

12.29.1 Detailed Description

Contains all information about a failed Move.

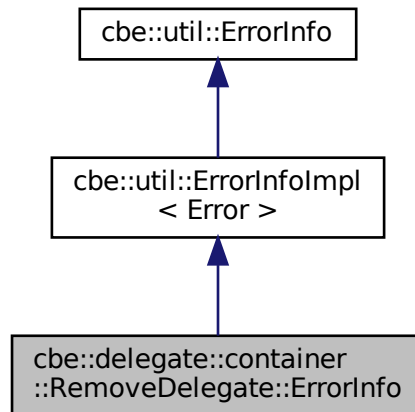
The documentation for this struct was generated from the following file:

- include/cbe/delegate/container/MoveDelegate.h

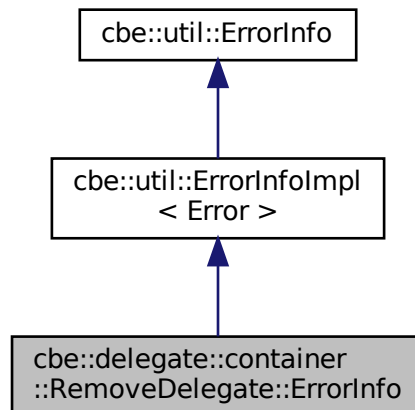
12.30 cbe::delegate::container::RemoveDelegate::ErrorInfo Struct Reference

```
#include <RemoveDelegate.h>
```

Inheritance diagram for cbe::delegate::container::RemoveDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::container::RemoveDelegate::ErrorInfo:



Additional Inherited Members

12.30.1 Detailed Description

Contains all information about a failed Remove.

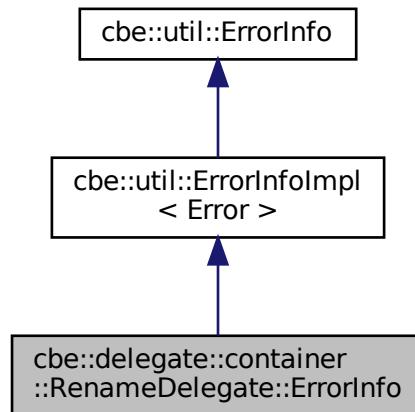
The documentation for this struct was generated from the following file:

- include/cbe/delegate/container/RemoveDelegate.h

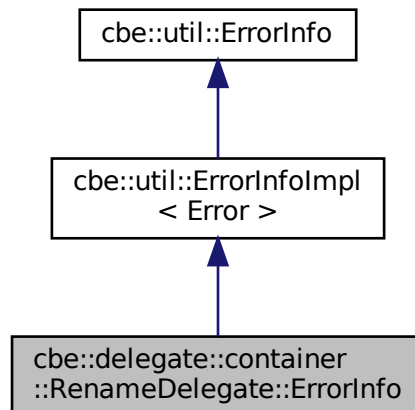
12.31 cbe::delegate::container::RenameDelegate::ErrorInfo Struct Reference

```
#include <RenameDelegate.h>
```

Inheritance diagram for cbe::delegate::container::RenameDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::container::RenameDelegate::ErrorInfo:



Additional Inherited Members

12.31.1 Detailed Description

Contains all information about a failed Rename.

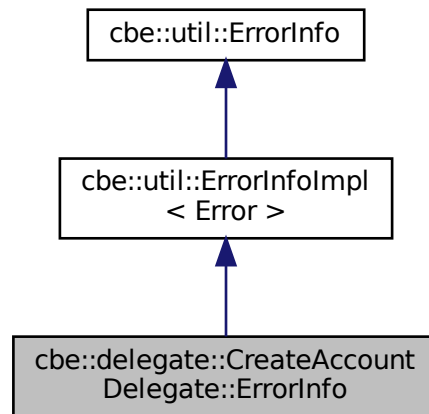
The documentation for this struct was generated from the following file:

- include/cbe/delegate/container/RenameDelegate.h

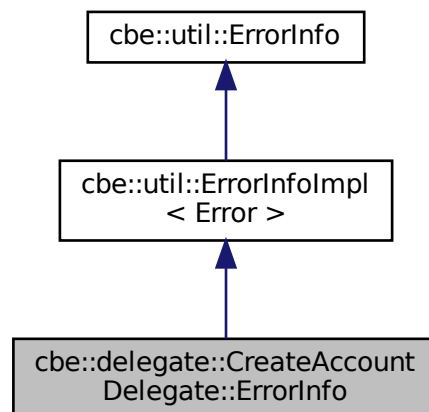
12.32 cbe::delegate::CreateAccountDelegate::ErrorInfo Struct Reference

```
#include <CreateAccountDelegate.h>
```

Inheritance diagram for cbe::delegate::CreateAccountDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::CreateAccountDelegate::ErrorInfo:



Additional Inherited Members

12.32.1 Detailed Description

Contains all information about a failed `CreateAccount`.

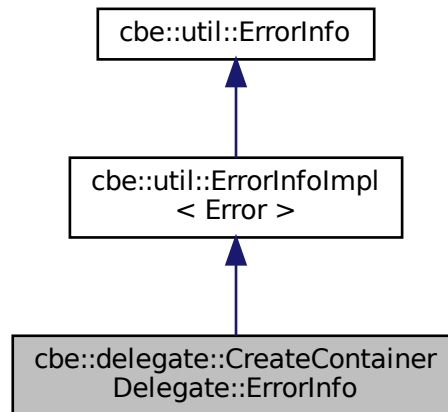
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/CreateAccountDelegate.h`

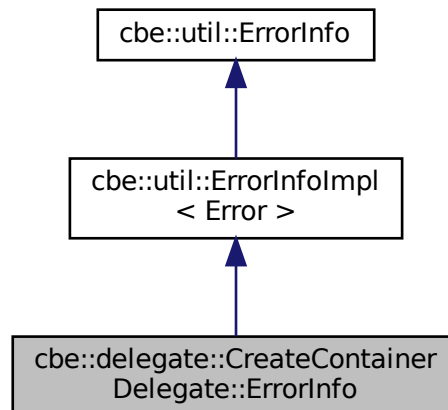
12.33 cbe::delegate::CreateContainerDelegate::ErrorInfo Struct Reference

```
#include <CreateContainerDelegate.h>
```

Inheritance diagram for cbe::delegate::CreateContainerDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::CreateContainerDelegate::ErrorInfo:



Additional Inherited Members

12.33.1 Detailed Description

Contains all information about a failed CreateContainer.

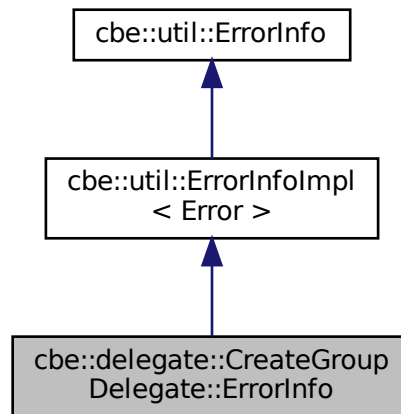
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/CreateContainerDelegate.h`

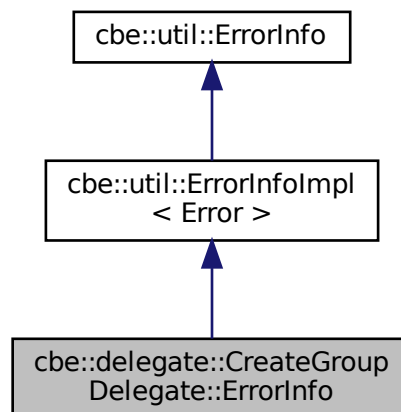
12.34 cbe::delegate::CreateGroupDelegate::ErrorInfo Struct Reference

```
#include <CreateGroupDelegate.h>
```

Inheritance diagram for cbe::delegate::CreateGroupDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::CreateGroupDelegate::ErrorInfo:



Additional Inherited Members

12.34.1 Detailed Description

Contains all information about a failed CreateGroup.

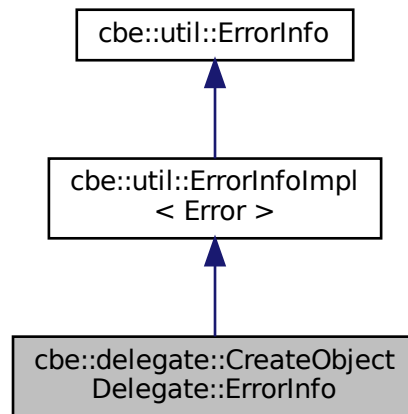
The documentation for this struct was generated from the following file:

- include/cbe/delegate/CreateGroupDelegate.h

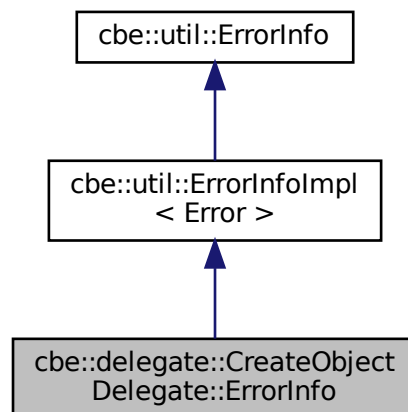
12.35 cbe::delegate::CreateObjectDelegate::ErrorInfo Struct Reference

```
#include <CreateObjectDelegate.h>
```

Inheritance diagram for cbe::delegate::CreateObjectDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::CreateObjectDelegate::ErrorInfo:



Additional Inherited Members

12.35.1 Detailed Description

Contains all information about a failed CreateObject.

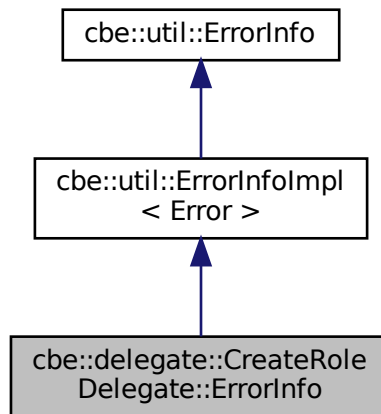
The documentation for this struct was generated from the following file:

- include/cbe/delegate/CreateObjectDelegate.h

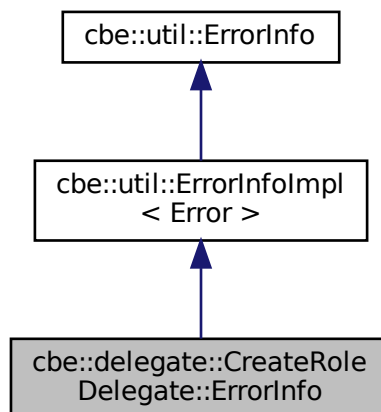
12.36 cbe::delegate::CreateRoleDelegate::ErrorInfo Struct Reference

```
#include <CreateRoleDelegate.h>
```

Inheritance diagram for cbe::delegate::CreateRoleDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::CreateRoleDelegate::ErrorInfo:



Additional Inherited Members

12.36.1 Detailed Description

Contains all information about a failed CreateRole.

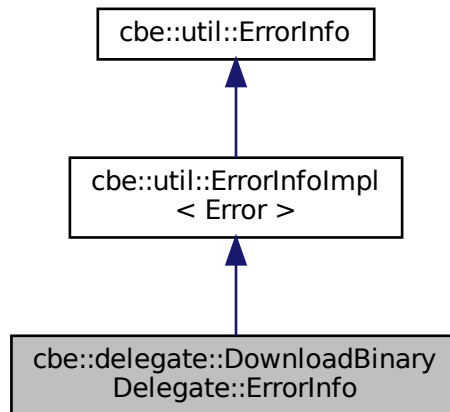
The documentation for this struct was generated from the following file:

- include/cbe/delegate/CreateRoleDelegate.h

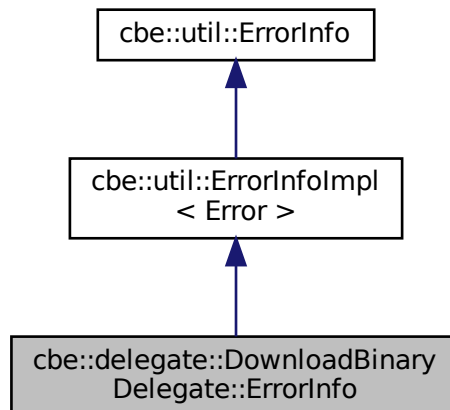
12.37 cbe::delegate::DownloadBinaryDelegate::ErrorInfo Struct Reference

```
#include <DownloadBinaryDelegate.h>
```

Inheritance diagram for cbe::delegate::DownloadBinaryDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::DownloadBinaryDelegate::ErrorInfo:



Additional Inherited Members

12.37.1 Detailed Description

Contains all information about a failed DownloadBinary.

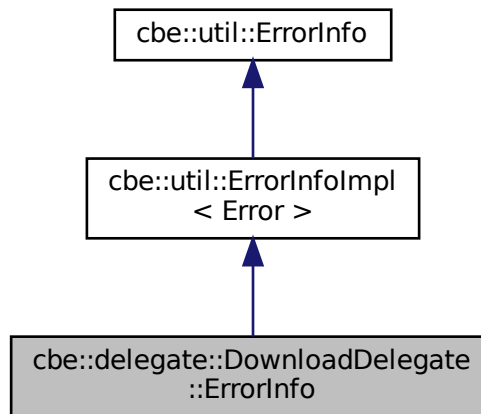
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/DownloadBinaryDelegate.h`

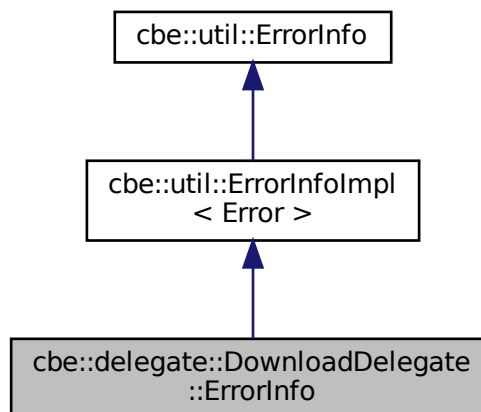
12.38 cbe::delegate::DownloadDelegate::ErrorInfo Struct Reference

```
#include <DownloadDelegate.h>
```

Inheritance diagram for cbe::delegate::DownloadDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::DownloadDelegate::ErrorInfo:



Additional Inherited Members

12.38.1 Detailed Description

Contains all information about a failed download.

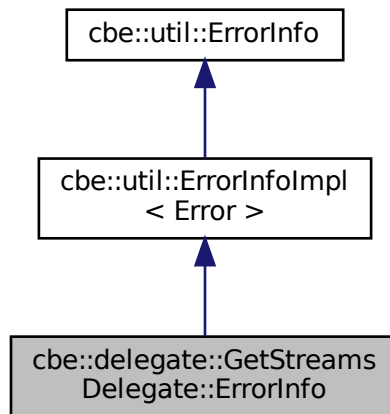
The documentation for this struct was generated from the following file:

- include/cbe/delegate/DownloadDelegate.h

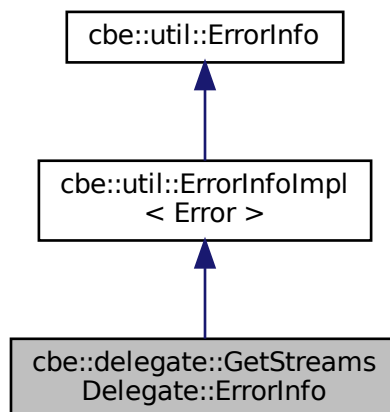
12.39 cbe::delegate::GetStreamsDelegate::ErrorInfo Struct Reference

```
#include <GetStreamsDelegate.h>
```

Inheritance diagram for cbe::delegate::GetStreamsDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::GetStreamsDelegate::ErrorInfo:



Additional Inherited Members

12.39.1 Detailed Description

Contains all information about a failed GetStreams.

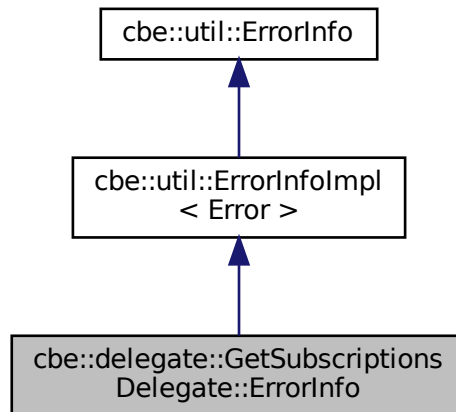
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/GetStreamsDelegate.h`

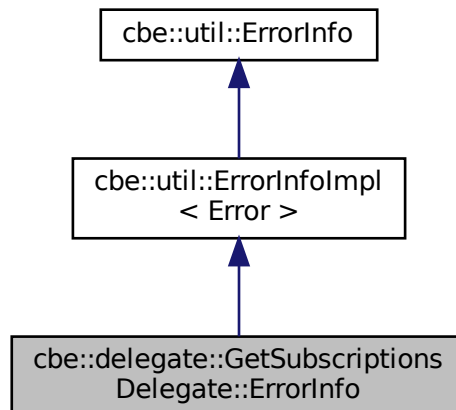
12.40 cbe::delegate::GetSubscriptionsDelegate::ErrorInfo Struct Reference

```
#include <GetSubscriptionsDelegate.h>
```

Inheritance diagram for cbe::delegate::GetSubscriptionsDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::GetSubscriptionsDelegate::ErrorInfo:



Additional Inherited Members

12.40.1 Detailed Description

Contains all information about a failed GetSubscriptions.

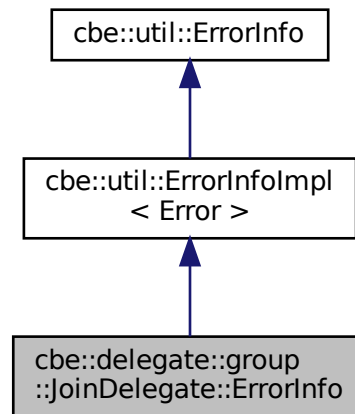
The documentation for this struct was generated from the following file:

- include/cbe/delegate/GetSubscriptionsDelegate.h

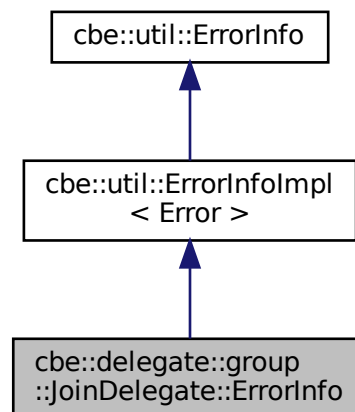
12.41 cbe::delegate::group::JoinDelegate::ErrorInfo Struct Reference

```
#include <JoinDelegate.h>
```

Inheritance diagram for cbe::delegate::group::JoinDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::group::JoinDelegate::ErrorInfo:



Additional Inherited Members

12.41.1 Detailed Description

Contains all information about a failed Join.

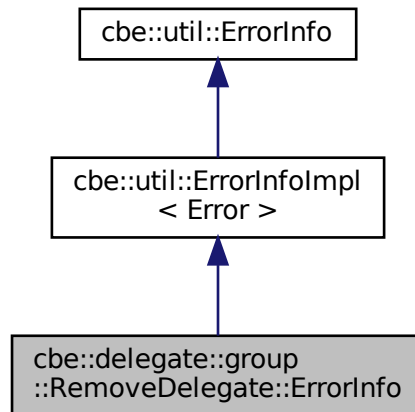
The documentation for this struct was generated from the following file:

- include/cbe/delegate/group/JoinDelegate.h

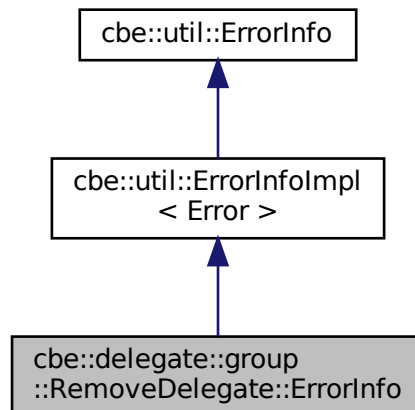
12.42 cbe::delegate::group::RemoveDelegate::ErrorInfo Struct Reference

```
#include <RemoveDelegate.h>
```

Inheritance diagram for cbe::delegate::group::RemoveDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::group::RemoveDelegate::ErrorInfo:



Additional Inherited Members

12.42.1 Detailed Description

Contains all information about a failed Remove.

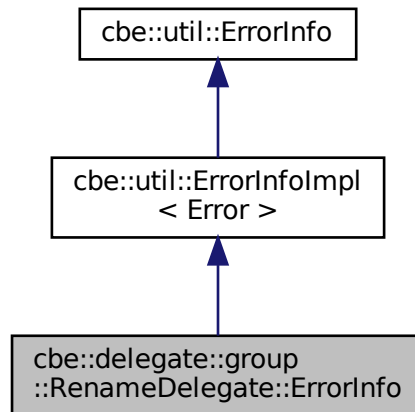
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/group/RemoveDelegate.h`

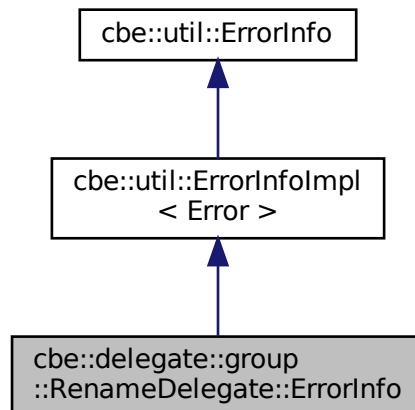
12.43 cbe::delegate::group::RenameDelegate::ErrorInfo Struct Reference

```
#include <RenameDelegate.h>
```

Inheritance diagram for cbe::delegate::group::RenameDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::group::RenameDelegate::ErrorInfo:



Additional Inherited Members

12.43.1 Detailed Description

Contains all information about a failed Rename.

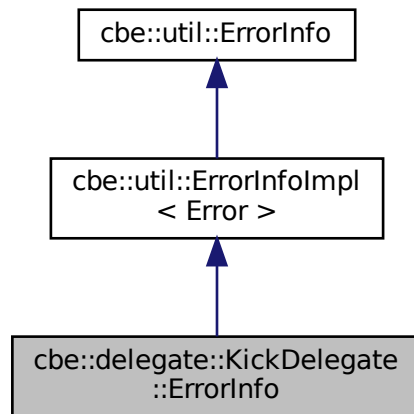
The documentation for this struct was generated from the following file:

- include/cbe/delegate/group/RenameDelegate.h

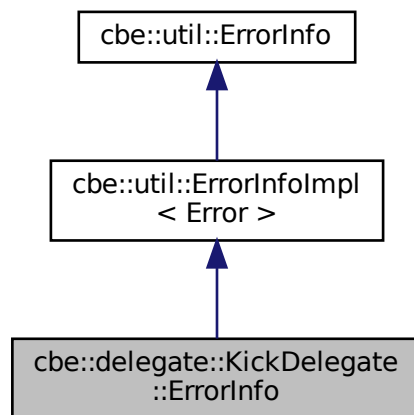
12.44 cbe::delegate::KickDelegate::ErrorInfo Struct Reference

```
#include <KickDelegate.h>
```

Inheritance diagram for cbe::delegate::KickDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::KickDelegate::ErrorInfo:



Additional Inherited Members

12.44.1 Detailed Description

Contains all information about a failed Kick.

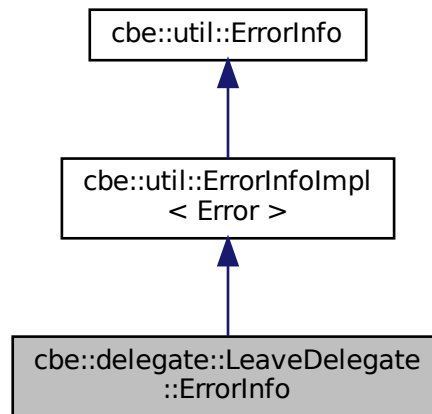
The documentation for this struct was generated from the following file:

- include/cbe/delegate/KickDelegate.h

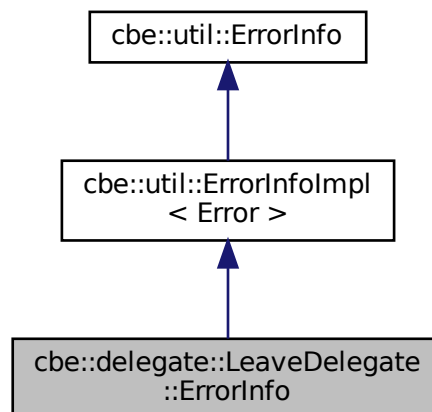
12.45 cbe::delegate::LeaveDelegate::ErrorInfo Struct Reference

```
#include <LeaveDelegate.h>
```

Inheritance diagram for cbe::delegate::LeaveDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::LeaveDelegate::ErrorInfo:



Additional Inherited Members

12.45.1 Detailed Description

Contains all information about a failed Leave.

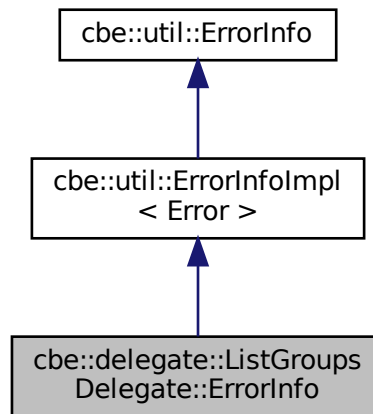
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/LeaveDelegate.h`

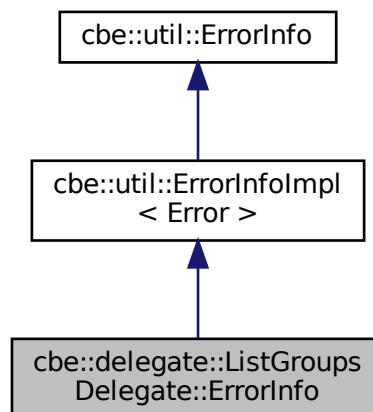
12.46 cbe::delegate::ListGroupsDelegate::ErrorInfo Struct Reference

```
#include <ListGroupsDelegate.h>
```

Inheritance diagram for cbe::delegate::ListGroupsDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::ListGroupsDelegate::ErrorInfo:



Additional Inherited Members

12.46.1 Detailed Description

Contains all information about a failed ListGroup.

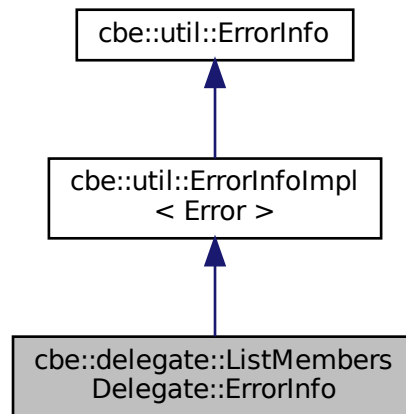
The documentation for this struct was generated from the following file:

- include/cbe/delegate/ListGroupsDelegate.h

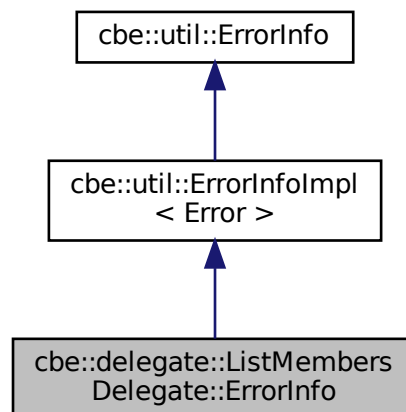
12.47 cbe::delegate::ListMembersDelegate::ErrorInfo Struct Reference

```
#include <ListMembersDelegate.h>
```

Inheritance diagram for cbe::delegate::ListMembersDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::ListMembersDelegate::ErrorInfo:



Additional Inherited Members

12.47.1 Detailed Description

Contains all information about a failed ListMembers.

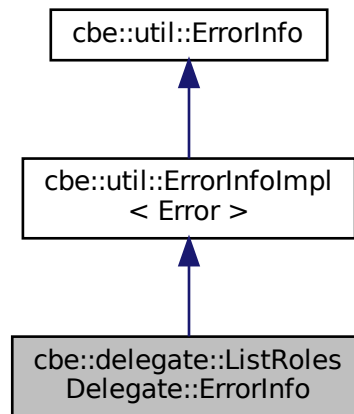
The documentation for this struct was generated from the following file:

- include/cbe/delegate/ListMembersDelegate.h

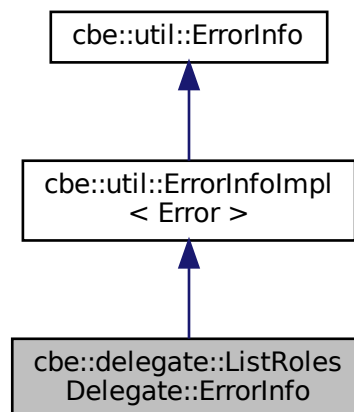
12.48 cbe::delegate::ListRolesDelegate::ErrorInfo Struct Reference

```
#include <ListRolesDelegate.h>
```

Inheritance diagram for cbe::delegate::ListRolesDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::ListRolesDelegate::ErrorInfo:



Additional Inherited Members

12.48.1 Detailed Description

Contains all information about a failed ListRoles.

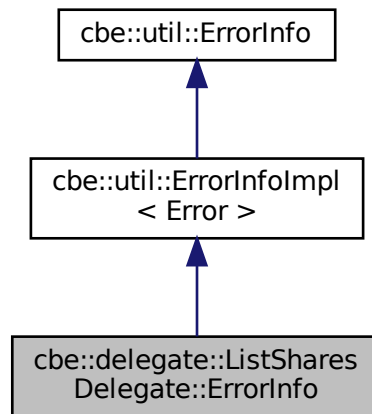
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/ListRolesDelegate.h`

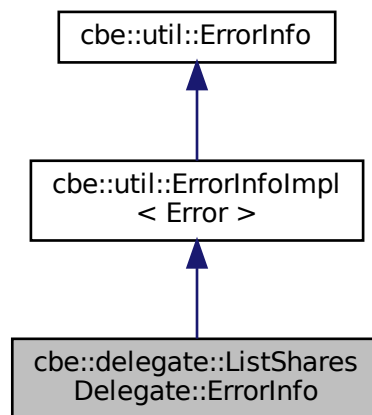
12.49 cbe::delegate::ListSharesDelegate::ErrorInfo Struct Reference

```
#include <ListSharesDelegate.h>
```

Inheritance diagram for cbe::delegate::ListSharesDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::ListSharesDelegate::ErrorInfo:



Additional Inherited Members

12.49.1 Detailed Description

Contains all information about a failed Share.

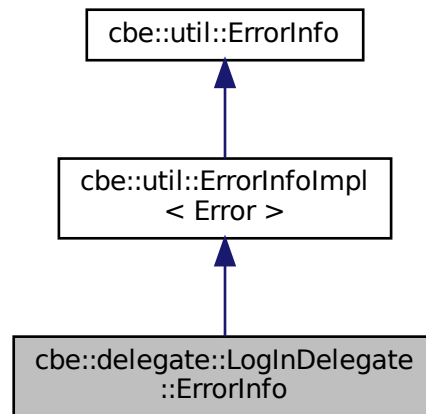
The documentation for this struct was generated from the following file:

- include/cbe/delegate/ListSharesDelegate.h

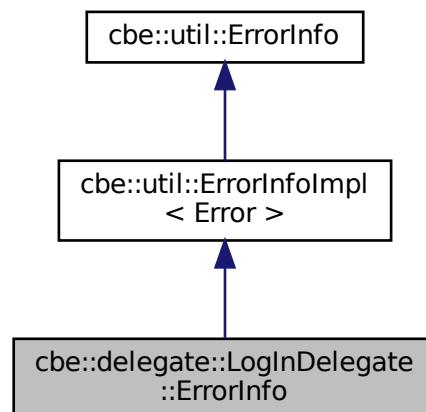
12.50 cbe::delegate::LogInDelegate::ErrorInfo Struct Reference

```
#include <LogInDelegate.h>
```

Inheritance diagram for cbe::delegate::LogInDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::LogInDelegate::ErrorInfo:



Additional Inherited Members

12.50.1 Detailed Description

Contains all information about a failed login.

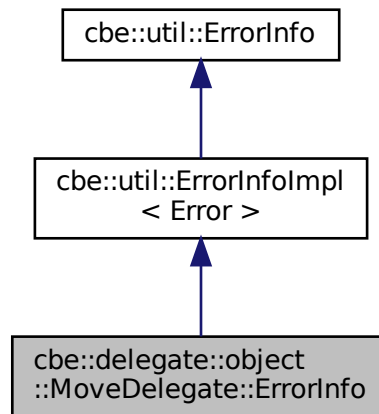
The documentation for this struct was generated from the following file:

- include/cbe/delegate/LogInDelegate.h

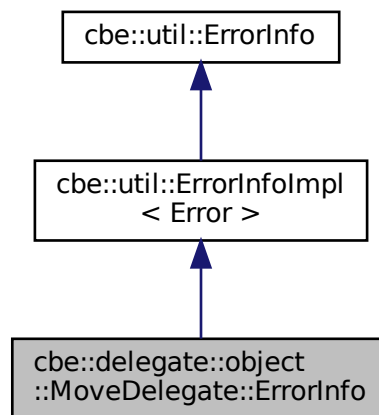
12.51 cbe::delegate::object::MoveDelegate::ErrorInfo Struct Reference

```
#include <MoveDelegate.h>
```

Inheritance diagram for cbe::delegate::object::MoveDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::object::MoveDelegate::ErrorInfo:



Additional Inherited Members

12.51.1 Detailed Description

Contains all information about a failed Move.

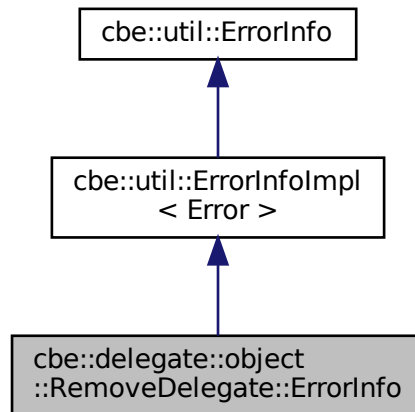
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/object/MoveDelegate.h`

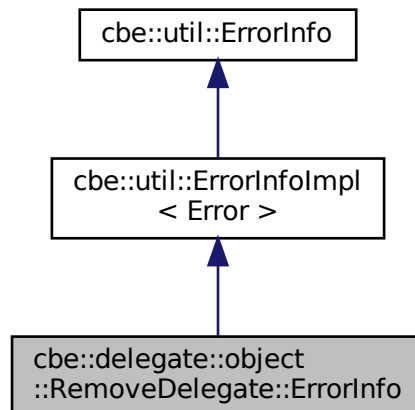
12.52 cbe::delegate::object::RemoveDelegate::ErrorInfo Struct Reference

```
#include <RemoveDelegate.h>
```

Inheritance diagram for cbe::delegate::object::RemoveDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::object::RemoveDelegate::ErrorInfo:



Additional Inherited Members

12.52.1 Detailed Description

Contains all information about a failed Remove.

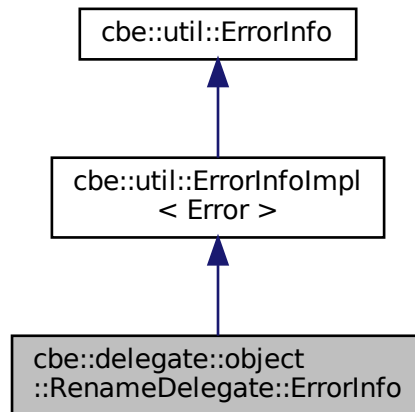
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/object/RemoveDelegate.h`

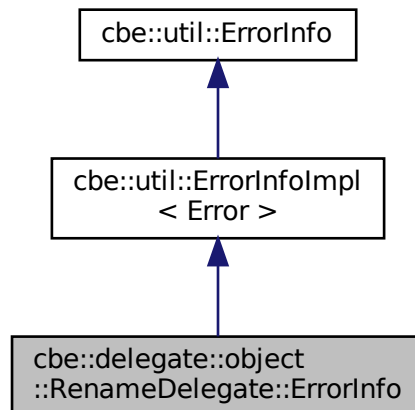
12.53 cbe::delegate::object::RenameDelegate::ErrorInfo Struct Reference

```
#include <RenameDelegate.h>
```

Inheritance diagram for cbe::delegate::object::RenameDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::object::RenameDelegate::ErrorInfo:



Additional Inherited Members

12.53.1 Detailed Description

Contains all information about a failed Rename.

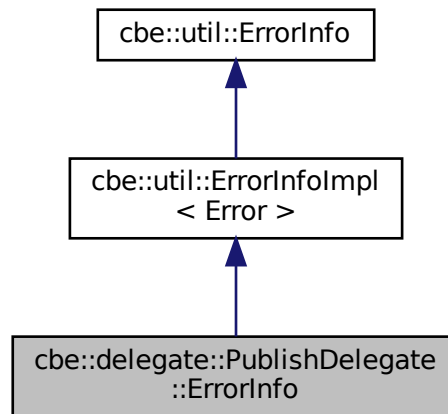
The documentation for this struct was generated from the following file:

- include/cbe/delegate/object/RenameDelegate.h

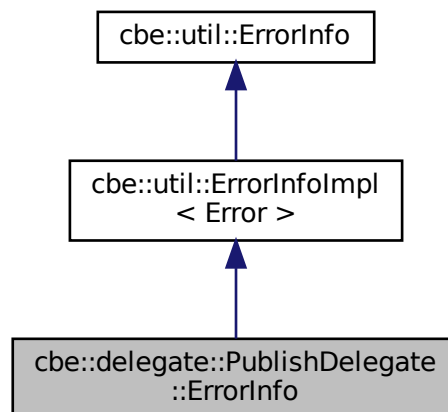
12.54 cbe::delegate::PublishDelegate::ErrorInfo Struct Reference

```
#include <PublishDelegate.h>
```

Inheritance diagram for cbe::delegate::PublishDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::PublishDelegate::ErrorInfo:



Additional Inherited Members

12.54.1 Detailed Description

Contains all information about a failed [Publish](#).

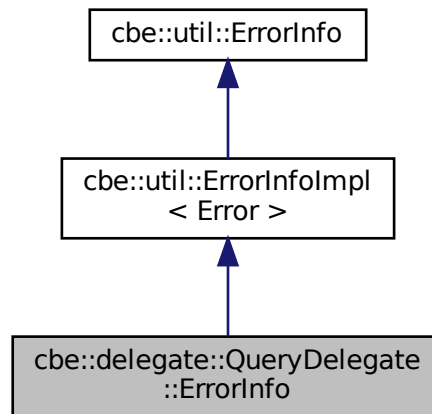
The documentation for this struct was generated from the following file:

- include/cbe/delegate/PublishDelegate.h

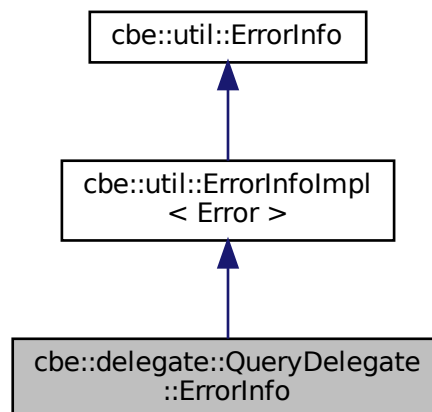
12.55 cbe::delegate::QueryDelegate::ErrorInfo Struct Reference

```
#include <QueryDelegate.h>
```

Inheritance diagram for cbe::delegate::QueryDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::QueryDelegate::ErrorInfo:



Additional Inherited Members

12.55.1 Detailed Description

Contains all information about a failed query.

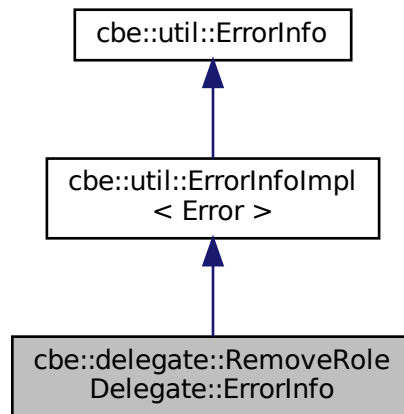
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/QueryDelegate.h`

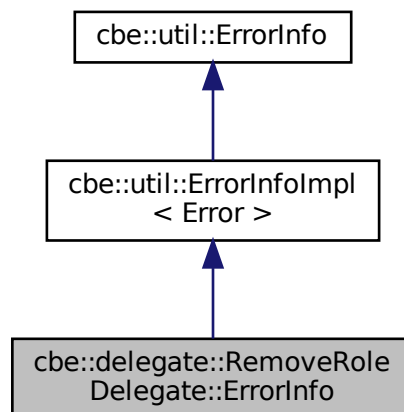
12.56 cbe::delegate::RemoveRoleDelegate::ErrorInfo Struct Reference

```
#include <RemoveRoleDelegate.h>
```

Inheritance diagram for cbe::delegate::RemoveRoleDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::RemoveRoleDelegate::ErrorInfo:



Additional Inherited Members

12.56.1 Detailed Description

Contains all information about a failed RemoveRole.

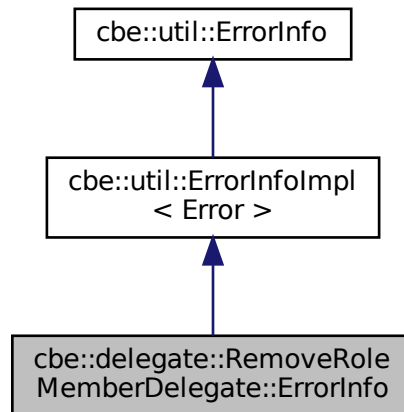
The documentation for this struct was generated from the following file:

- include/cbe/delegate/RemoveRoleDelegate.h

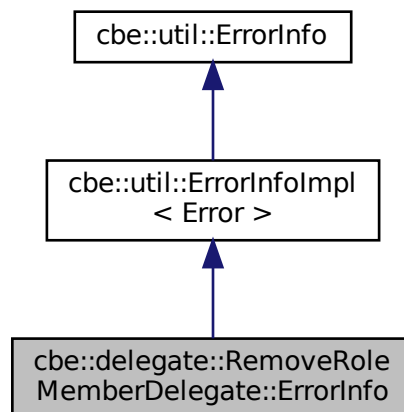
12.57 cbe::delegate::RemoveRoleMemberDelegate::ErrorInfo Struct Reference

```
#include <RemoveRoleMemberDelegate.h>
```

Inheritance diagram for cbe::delegate::RemoveRoleMemberDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::RemoveRoleMemberDelegate::ErrorInfo:



Additional Inherited Members

12.57.1 Detailed Description

Contains all information about a failed RemoveMember.

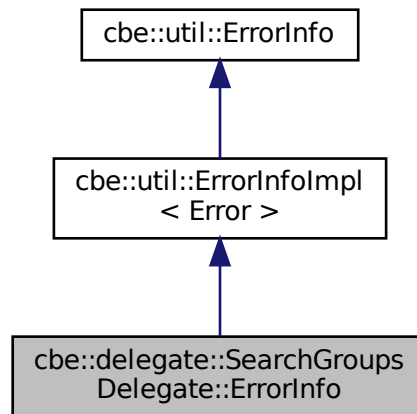
The documentation for this struct was generated from the following file:

- `include/cbe/delegate/RemoveRoleMemberDelegate.h`

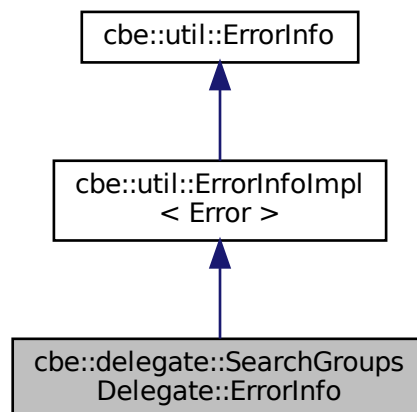
12.58 cbe::delegate::SearchGroupsDelegate::ErrorInfo Struct Reference

```
#include <SearchGroupsDelegate.h>
```

Inheritance diagram for cbe::delegate::SearchGroupsDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::SearchGroupsDelegate::ErrorInfo:



Additional Inherited Members

12.58.1 Detailed Description

Contains all information about a failed SearchGroups.

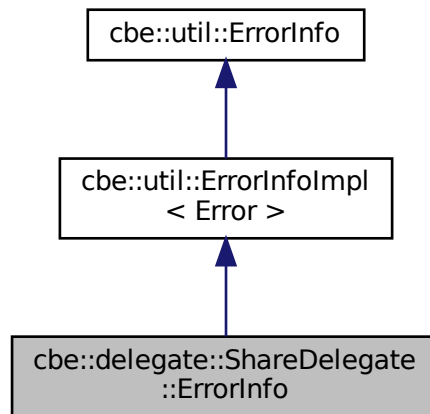
The documentation for this struct was generated from the following file:

- include/cbe/delegate/SearchGroupsDelegate.h

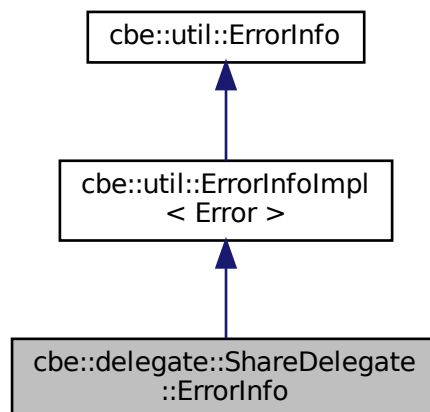
12.59 cbe::delegate::ShareDelegate::ErrorInfo Struct Reference

```
#include <ShareDelegate.h>
```

Inheritance diagram for cbe::delegate::ShareDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::ShareDelegate::ErrorInfo:



Additional Inherited Members

12.59.1 Detailed Description

Contains all information about a failed Share.

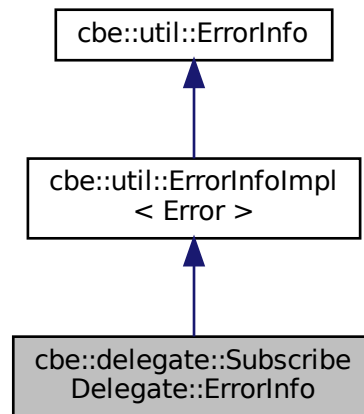
The documentation for this struct was generated from the following file:

- include/cbe/delegate/ShareDelegate.h

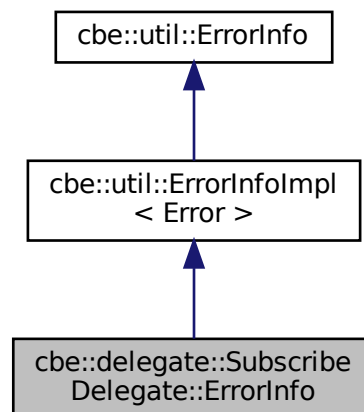
12.60 cbe::delegate::SubscribeDelegate::ErrorInfo Struct Reference

```
#include <SubscribeDelegate.h>
```

Inheritance diagram for cbe::delegate::SubscribeDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::SubscribeDelegate::ErrorInfo:



Additional Inherited Members

12.60.1 Detailed Description

Contains all information about a failed subscribe.

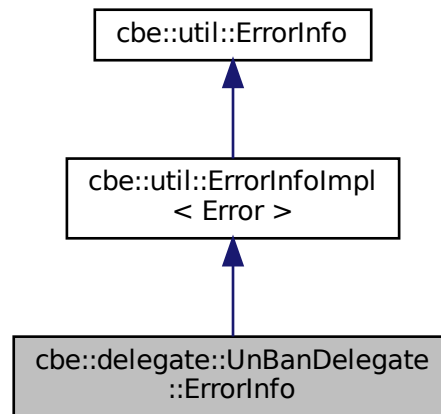
The documentation for this struct was generated from the following file:

- include/cbe/delegate/SubscribeDelegate.h

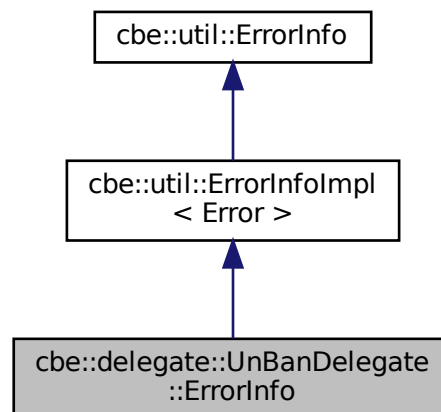
12.61 cbe::delegate::UnBanDelegate::ErrorInfo Struct Reference

```
#include <UnBanDelegate.h>
```

Inheritance diagram for cbe::delegate::UnBanDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::UnBanDelegate::ErrorInfo:



Additional Inherited Members

12.61.1 Detailed Description

Contains all information about a failed UnBan.

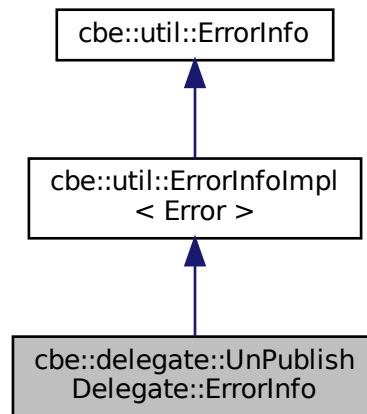
The documentation for this struct was generated from the following file:

- include/cbe/delegate/UnBanDelegate.h

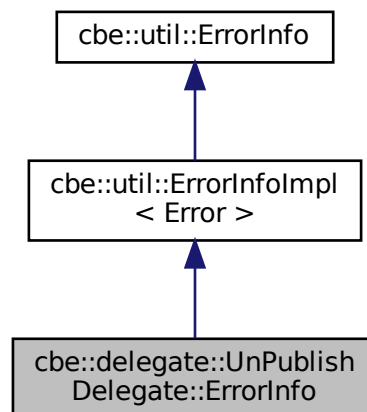
12.62 cbe::delegate::UnPublishDelegate::ErrorInfo Struct Reference

```
#include <UnPublishDelegate.h>
```

Inheritance diagram for cbe::delegate::UnPublishDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::UnPublishDelegate::ErrorInfo:



Additional Inherited Members

12.62.1 Detailed Description

Contains all information about a failed UnPublish.

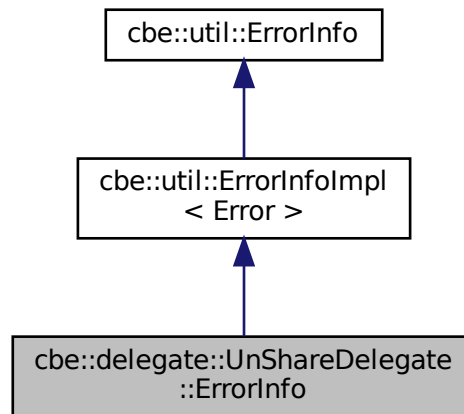
The documentation for this struct was generated from the following file:

- include/cbe/delegate/UnPublishDelegate.h

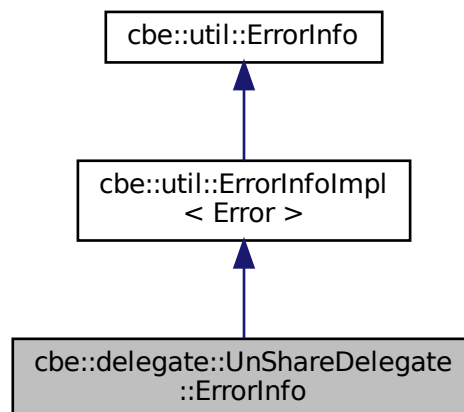
12.63 cbe::delegate::UnShareDelegate::ErrorInfo Struct Reference

```
#include <UnShareDelegate.h>
```

Inheritance diagram for cbe::delegate::UnShareDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::UnShareDelegate::ErrorInfo:



Additional Inherited Members

12.63.1 Detailed Description

Contains all information about a failed UnShare.

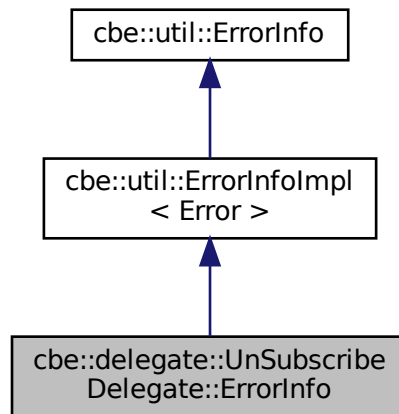
The documentation for this struct was generated from the following file:

- include/cbe/delegate/UnShareDelegate.h

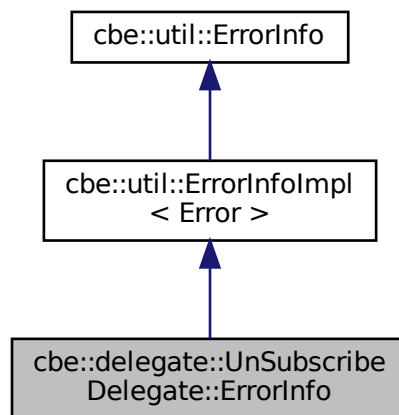
12.64 cbe::delegate::UnSubscribeDelegate::ErrorInfo Struct Reference

```
#include <UnSubscribeDelegate.h>
```

Inheritance diagram for cbe::delegate::UnSubscribeDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::UnSubscribeDelegate::ErrorInfo:



Additional Inherited Members

12.64.1 Detailed Description

Contains all information about a failed UnSubscribe.

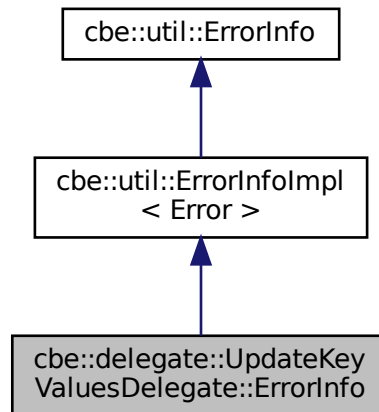
The documentation for this struct was generated from the following file:

- include/cbe/delegate/UnSubscribeDelegate.h

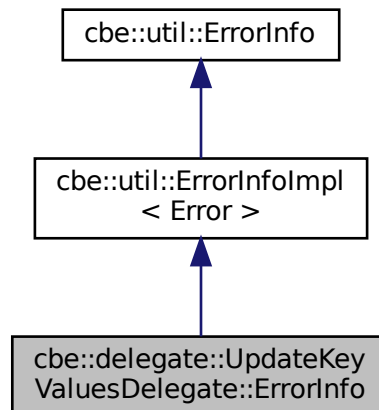
12.65 cbe::delegate::UpdateKeyValuesDelegate::ErrorInfo Struct Reference

```
#include <UpdateKeyValuesDelegate.h>
```

Inheritance diagram for cbe::delegate::UpdateKeyValuesDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::UpdateKeyValuesDelegate::ErrorInfo:



Additional Inherited Members

12.65.1 Detailed Description

Contains all information about a failed UpdateKeyValues.

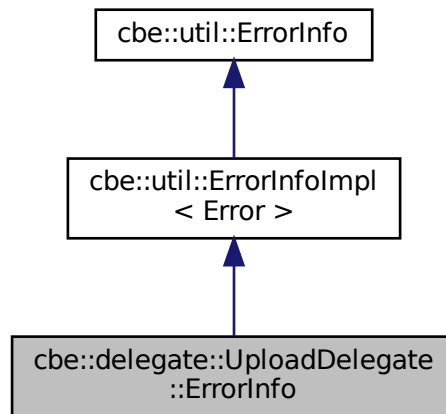
The documentation for this struct was generated from the following file:

- include/cbe/delegate/UpdateKeyValuesDelegate.h

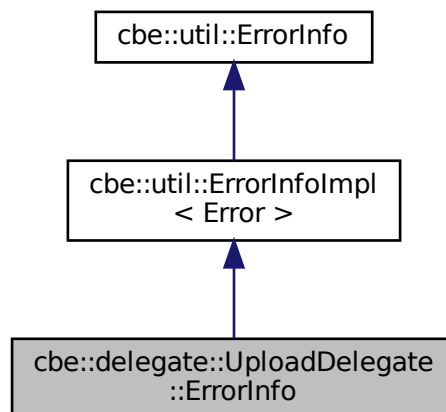
12.66 cbe::delegate::UploadDelegate::ErrorInfo Struct Reference

```
#include <UploadDelegate.h>
```

Inheritance diagram for cbe::delegate::UploadDelegate::ErrorInfo:



Collaboration diagram for cbe::delegate::UploadDelegate::ErrorInfo:



Additional Inherited Members

12.66.1 Detailed Description

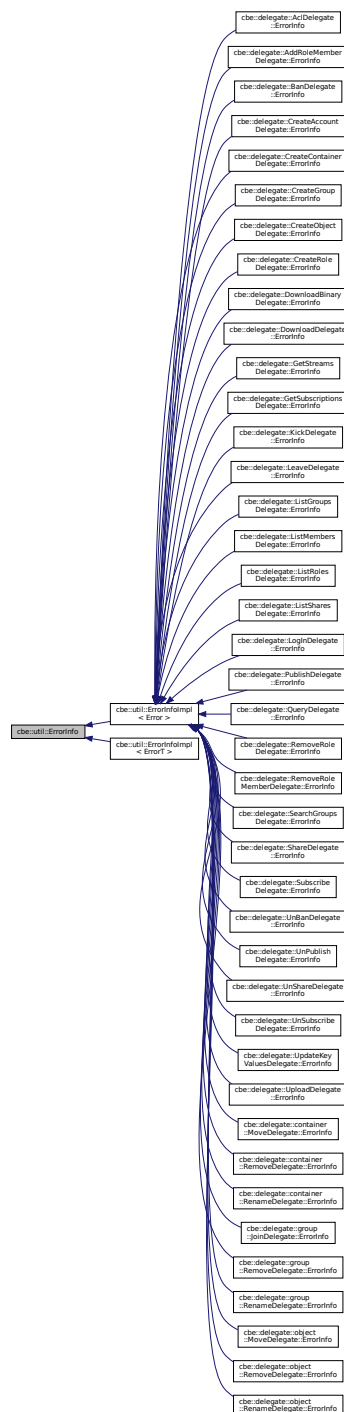
Contains all information about a failed Upload.

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

- include/cbe/delegate/UploadDelegate.h

12.67 cbe::util::ErrorInfo Struct Reference

Inheritance diagram for cbe::util::ErrorInfo:



Public Member Functions

- virtual void **printError** (std::ostream &os) const =0
- **operator bool** () const

Public Attributes

- std::string **contextStr** {}

Protected Member Functions

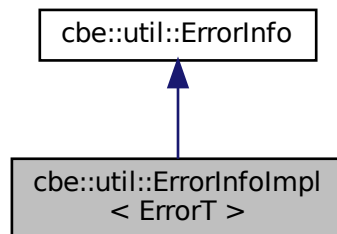
- **ErrorInfo** ([Context](#) &&context)

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

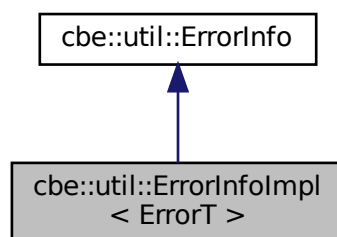
- include/cbe/util/ErrorInfo.h

12.68 cbe::util::ErrorInfoImpl< ErrorT > Struct Template Reference

Inheritance diagram for cbe::util::ErrorInfoImpl< ErrorT >:



Collaboration diagram for cbe::util::ErrorInfoImpl< ErrorT >:



Public Types

- using **Base** = [ErrorInfoImpl](#)< ErrorT >

Public Member Functions

- **ErrorInfoImpl** ([Context](#) &&context, ErrorT &&error)
- template<class ErrorT2 >
[ErrorInfoImpl](#) & **operator=** ([ErrorInfoImpl](#)< ErrorT2 > &&rh)

- void **printError** (std::ostream &os) const final
- template<class ErrorT2 >
[ErrorInfoImpl](#)< ErrorT > & **operator=** ([ErrorInfoImpl](#)< ErrorT2 > &&rh)

Public Attributes

- ErrorT **error** {}

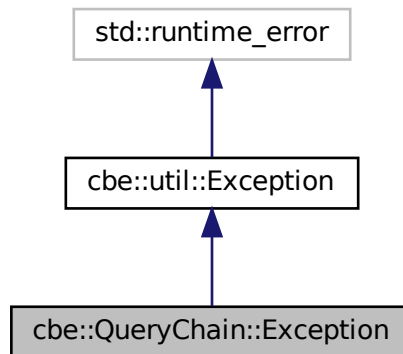
Additional Inherited Members

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

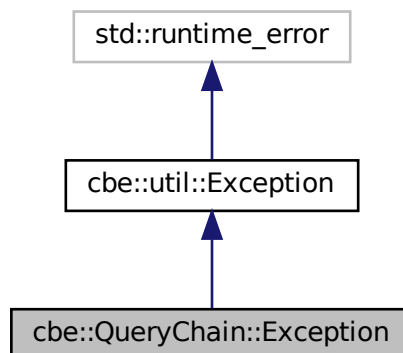
- include/cbe/util/ErrorInfo.h

12.69 cbe::QueryChain::Exception Struct Reference

Inheritance diagram for cbe::QueryChain::Exception:



Collaboration diagram for cbe::QueryChain::Exception:



Additional Inherited Members

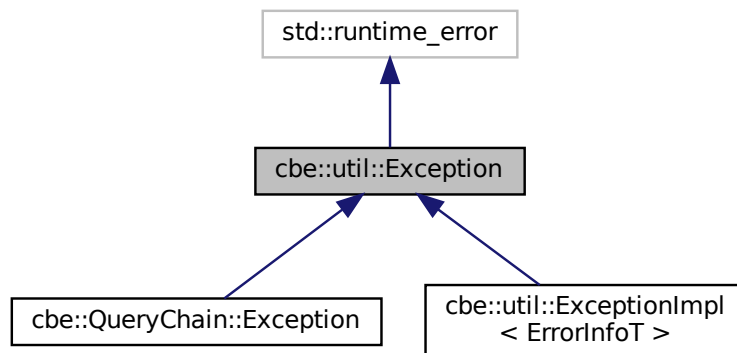
The documentation for this struct was generated from the following file:

- include/cbe/QueryChain.h

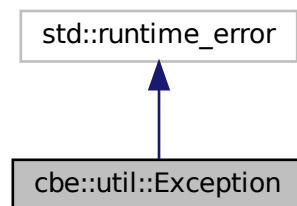
12.70 cbe::util::Exception Struct Reference

```
#include <Exception.h>
```

Inheritance diagram for cbe::util::Exception:



Collaboration diagram for cbe::util::Exception:



Public Member Functions

- `template<typename... Ts>`
Exception (Ts &&... args)
- `std::string typeAsString () const`
- `template<class CbeExceptionT >`
`const CbeExceptionT * derivedCbeException () const`
Examines whether current exception is of type CbeExceptionT.

12.70.1 Detailed Description

X

12.70.2 Member Function Documentation

12.70.2.1 derivedCbeException()

```
template<class CbeExceptionT >
const CbeExceptionT* cbe::util::Exception::derivedCbeException ( ) const [inline]
Examines whether current exception is of type CbeExceptionT.
```

Template Parameters

<i>CbeExceptionT</i>	Subtype of cbe::util::Exception to be examined.
----------------------	---

Returns

Pointer to the requested exception, nullptr implies no such subclass type.

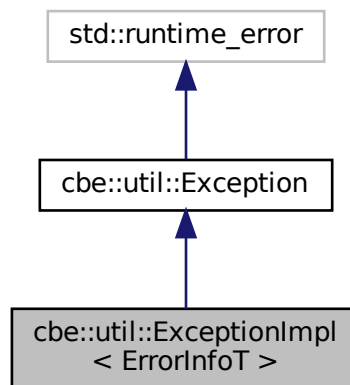
The documentation for this struct was generated from the following file:

- include/cbe/util/Exception.h

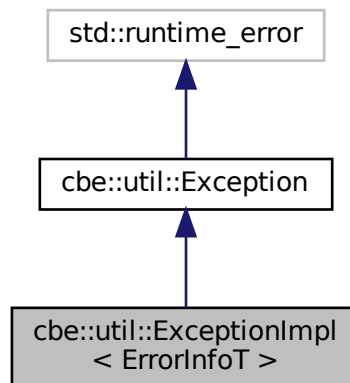
12.71 cbe::util::ExceptionImpl< ErrorInfoT > Struct Template Reference

```
#include <Exception.h>
```

Inheritance diagram for cbe::util::ExceptionImpl< ErrorInfoT >:



Collaboration diagram for `cbe::util::ExceptionImpl< ErrorInfoT >`:



Public Types

- using **Base** = [ExceptionImpl](#)< ErrorInfoT >

Public Member Functions

- **ExceptionImpl** (ErrorInfoT &&errorInfo)

Public Attributes

- const ErrorInfoT **errorInfo**

12.71.1 Detailed Description

```
template<class ErrorInfoT>
struct cbe::util::ExceptionImpl< ErrorInfoT >
```

XX

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

- include/cbe/util/Exception.h

12.72 cbe::Filter Class Reference

Use to select [Item](#) that meets specific criterias when doing a query.

```
#include <Filter.h>
```

Public Member Functions

- **Filter** (const [Filter](#) &rh)
- **Filter** ([Filter](#) &&rh)
- [Filter](#) & **operator=** (const [Filter](#) &rh)
- [Filter](#) & **operator=** ([Filter](#) &&rh)
- [cbe::ItemType](#) **getDataType** () const

Returns the requested data type.

- `std::string getQuery () const`
Returns the query string that was set on the filter.
- `bool getAscending () const`
Returns the settings on how the data was sorted and displayed.
- `bool getDeleted () const`
Determines if deleted objects in the bin are included.
- `bool getContainerFirst () const`
Determines if containers should be sorted before objects.
- `bool isJoinedResult () const`
Determines if the resultset comes from a joined query.
- `std::uint32_t getOffset () const`
Determines the offset of items to be skipped.
- `std::uint32_t getCount () const`
Returns the size of the resultset of the query.
- `bool getByPassCache () const`
Returns if skipping the cache was used or not.
- `cbe::FilterOrder getItemOrder () const`
Returns the order the query was sorted.
- `cbe::Container container ()`
Returns the parent container that was queried, if it is in the cache. This is after a query was done, not before.
- `cbe::Filter & setDataTypes (cbe::ItemType itemType)`
Set which datatypes to query for.
- `cbe::Filter & setQuery (std::string query)`
Set the query string.
- `cbe::Filter & setAscending (bool ascending)`
*Sets the order, *ascending*, in which data should be displayed.*
- `cbe::Filter & setDeleted (bool deleted)`
Whether to include deleted items.
- `cbe::Filter & setContainerFirst (bool containerFirst)`
*If *Container* should be displayed before *Object*.*
- `cbe::Filter & setOffset (std::uint32_t offset)`
Set the offset for paging.
- `cbe::Filter & setCount (std::uint32_t count)`
Set the maximum resultset.
- `cbe::Filter & setItemOrder (cbe::FilterOrder order)`
*Set the order of how data will be shown by the enum of *FilterOrder*.*
- `cbe::Filter & setByPassCache (bool byPassCache)`
Set to ignore the local SDK cache.

Friends

- class **CloudBackend**
- class **Container**
- class **QueryChain**
- class **QueryResult**
- `std::ostream & operator<< (std::ostream &os, const Filter &filter)`
- `CBI::ItemDelegatePtr delegate::createCbiQueryDelegate (delegate::QueryDelegatePtr, cbe::util::Context &&)`
- `CBI::ItemDelegatePtr delegate::createCbiQueryDelegate (delegate::QueryJoinDelegatePtr, cbe::util::Context &&)`

12.72.1 Detailed Description

Use to select [Item](#) that meets specific criterias when doing a query.

Filtering on metadata KeyValues can be done with [setQuery\(\)](#) and only to query for [Object](#), not [Container](#).

12.72.2 Member Function Documentation

12.72.2.1 `getDataType()`

```
cbe::ItemType cbe::Filter::getDataType ( ) const
```

Returns the requested data type.

Such as `ItemType` e.g.,

- `cbe::ItemType::Container`
- `cbe::ItemType::Object`
- `cbe::ItemType::Group`

12.72.2.2 `getItemOrder()`

```
cbe::FilterOrder cbe::Filter::getItemOrder ( ) const
```

Returns the order the query was sorted.

Check enum `FilterOrder` to see options for sorting.

12.72.2.3 `getOffset()`

```
std::uint32_t cbe::Filter::getOffset ( ) const
```

Determines the offset of items to be skipped.

Returns the offset of items in the beginning of total resultset to be skipped, that was used for the filter in the query.

12.72.2.4 `getQuery()`

```
std::string cbe::Filter::getQuery ( ) const
```

Returns the query string that was set on the filter.

Query string e.g.,

- `name:Abc*`
- `age:100`
- `price<200`
- `size>40`

12.72.2.5 `setAscending()`

```
cbe::Filter& cbe::Filter::setAscending (
    bool ascending )
```

Sets the order, `ascending`, in which data should be displayed.

Parameters

<i>ascending</i>	true = sorted in increasing order false = sorted in decreasing order
------------------	---

12.72.2.6 setByPassCache()

```
cbe::Filter& cbe::Filter::setByPassCache (
    bool byPassCache )
```

Set to ignore the local SDK cache.

Instead allways fetch the resultset from the cloud.

Parameters

<i>byPassCache</i>	true = skip the SDK cache and ask the cloud. default is to use the cache when available.
--------------------	---

12.72.2.7 setContainerFirst()

```
cbe::Filter& cbe::Filter::setContainerFirst (
    bool containerFirst )
```

If [Container](#) should be displayed before [Object](#).

Parameters

<i>containerFirst</i>	true = sort containers at the top of the resultset. default is to not sort for this.
-----------------------	---

12.72.2.8 setCount()

```
cbe::Filter& cbe::Filter::setCount (
    std::uint32_t count )
```

Set the maximum resultset.

Set the maximum number of items you want to get from a container.

E.g.: a container has 50 items but you only want the 10 first in ascending order then set ascending to true and setCount to 10.

Parameters

<i>count</i>	maximum number of items to return
--------------	-----------------------------------

12.72.2.9 setDataType()

```
cbe::Filter& cbe::Filter::setDataType (
    cbe::ItemType itemType )
```

Set which datatypes to query for.

Parameters

<i>itemType</i>	as defined in Types.h
-----------------	---------------------------------------

12.72.2.10 setDeleted()

```
cbe::Filter& cbe::Filter::setDeleted (
    bool deleted )
```

Whether to include deleted items.

Parameters

<i>deleted</i>	true = include items in the bin default is false
----------------	---

12.72.2.11 setItemOrder()

```
cbe::Filter& cbe::Filter::setItemOrder (
    cbe::FilterOrder order )
```

Set the order of how data will be shown by the enum of FilterOrder.

E.g.: Title first, Relevance, published e.t.c.

Parameters

<i>order</i>	see the enum <code>FilterOrder</code> in Types.h
--------------	--

12.72.2.12 setOffset()

```
cbe::Filter& cbe::Filter::setOffset (
    std::uint32_t offset )
```

Set the offset for paging.

Offset is the item offset where to start your resultset.

E.g.: There is already a query resultset of the first 100 items and to get the rest you put [setOffset\(\)](#) to 100 to get the next resultset from 101-.

Parameters

<i>offset</i>	first item start number of the complete resultset to return
---------------	---

12.72.2.13 setQuery()

```
cbe::Filter& cbe::Filter::setQuery (
    std::string query )
```

Set the query string.

E.g., `firstname:*` (would search for all objects with the metadata key labeled `firstname`).

Search string e.g.,

- `name:Abc*`
- `age:100`
- `price<200`
- `size>40`

Note

If used with `rootContainer` id as the `dataId` to search in, the whole account will be searched.

Parameters

<i>query</i>	is a string of key tags or key:value pairs. E.g. Name:* Country:Sweden Country:Norway This will search for objects with key Name but any value and where key Country is either Sweden or Norway.
--------------	--

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

- include/cbe/Filter.h

12.73 cbe::delegate::GetStreamsDelegate Class Reference

```
#include <GetStreamsDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Streams](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onGetStreamsSuccess](#) ([cbe::Streams](#) &&streams)=0
- virtual void [onGetStreamsError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.73.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Object::getStreams\(\)](#)
See [Example of retrieving the Stream attached to a cbe::Object with Object::getStreams\(\)](#)

12.73.2 Member Function Documentation

12.73.2.1 onGetStreamsError()

```
virtual void cbe::delegate::GetStreamsDelegate::onGetStreamsError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.73.2.2 onGetStreamsSuccess()

```
virtual void cbe::delegate::GetStreamsDelegate::onGetStreamsSuccess (
    cbe::Streams && streams ) [pure virtual]
```

Called upon successful Object::GetStreams.

Parameters

<i>streams</i>	The currently loaded stream(s) attached to the requested object.
----------------	--

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

- include/cbe/delegate/GetStreamsDelegate.h

12.74 cbe::delegate::GetSubscriptionsDelegate Class Reference

```
#include <GetSubscriptionsDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::QueryResult](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onGetSubscriptionsSuccess](#) ([cbe::QueryResult](#) &&object)=0
- virtual void [onGetSubscriptionsError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.74.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::SubscribeManager::getSubscriptions](#)

12.74.2 Member Function Documentation

12.74.2.1 onGetSubscriptionsError()

```
virtual void cbe::delegate::GetSubscriptionsDelegate::onGetSubscriptionsError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.74.2.2 onGetSubscriptionsSuccess()

```
virtual void cbe::delegate::GetSubscriptionsDelegate::onGetSubscriptionsSuccess (
    cbe::QueryResult && object ) [pure virtual]
```

Called upon successful GetSubscriptions.

Parameters

<i>object</i>	Instance of object that is being streamed.
---------------	--

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

- include/cbe/delegate/GetSubscriptionsDelegate.h

12.75 cbe::Group Class Reference

A group of members.

```
#include <Group.h>
```

Public Types

- using **Requests** = std::vector< [cbe::Request](#) >
- using [CreateGroupDelegatePtr](#) = [delegate::CreateGroupDelegatePtr](#)
- using [JoinDelegatePtr](#) = [delegate::group::JoinDelegatePtr](#)
- using [LeaveDelegatePtr](#) = [delegate::LeaveDelegatePtr](#)
- using [RemoveDelegatePtr](#) = [delegate::group::RemoveDelegatePtr](#)
- using [RenameDelegatePtr](#) = [delegate::group::RenameDelegatePtr](#)
- using [ListMembersDelegatePtr](#) = [delegate::ListMembersDelegatePtr](#)
- using [ListBannedMembersDelegatePtr](#) = [delegate::ListMembersDelegatePtr](#)
- using [ListRolesDelegatePtr](#) = [delegate::ListRolesDelegatePtr](#)
- using [CreateRoleDelegatePtr](#) = [delegate::CreateRoleDelegatePtr](#)
- using [RemoveRoleDelegatePtr](#) = [delegate::RemoveRoleDelegatePtr](#)

Public Member Functions

- std::string [name](#) () const
- [cbe::GroupId](#) [id](#) () const
- [cbe::GroupId](#) [parentId](#) () const
- [cbe::Container](#) [groupContainer](#) () const
- [cbe::Visibility](#) [getVisibility](#) () const
- bool [joined](#) () const
- Requests [requests](#) () const
- [cbe::Group](#) [createGroup](#) (std::string [name](#), std::string [memberAlias](#), [cbe::Visibility](#) [visibility](#), [CreateGroupDelegatePtr](#) [delegate](#))
Create a new [Group](#).
- void [join](#) (std::string [alias](#), [cbe::Visibility](#) [memberVisibility](#), std::string [applicationComment](#), [JoinDelegatePtr](#) [delegate](#))
Synchronous [exception] Ask to join a group. In this first version All members will be Public, meaning visible for other member inside the group. All groups will also be open so all join requests should be accepted directly.
- void [leave](#) ([LeaveDelegatePtr](#) [delegate](#))
Leave group.
- void [remove](#) ([RemoveDelegatePtr](#) [delegate](#))
Remove group.
- void [rename](#) (std::string [newName](#), [RenameDelegatePtr](#) [delegate](#))
Rename the [Group](#); group id does not change.
- void [listMembers](#) ([ListMembersDelegatePtr](#) [delegate](#))
List all members in the group.
- void [listBannedMembers](#) ([ListBannedMembersDelegatePtr](#) [delegate](#))
Lists all banned former members, or users.
- void [listRoles](#) ([ListRolesDelegatePtr](#) [delegate](#))
Lists all roles in the group.
- void [createRole](#) (std::string [name](#), [CreateRoleDelegatePtr](#) [delegate](#))
Creates a new role in the group.
- void [removeRole](#) ([RoleId](#) [roleId](#), [RemoveRoleDelegatePtr](#) [delegate](#))
- [Group](#) ([cbe::DefaultCtor](#))
Default constructor.
- [operator bool](#) () const
Checks if the current instance is real.

Friends

- class **GroupManager**
- class **GroupQueryResult**

12.75.1 Detailed Description

A group of members.

12.75.2 Member Typedef Documentation

12.75.2.1 CreateGroupDelegatePtr

```
using cbe::Group::CreateGroupDelegatePtr = delegate::CreateGroupDelegatePtr
```

Pointer to CreateGroupDelegate that is passed into:

Group::createGroup(std::string,std::string,CreateGroupDelegatePtr,cbe::Visibility).

12.75.2.2 CreateRoleDelegatePtr

```
using cbe::Group::CreateRoleDelegatePtr = delegate::CreateRoleDelegatePtr
```

Pointer to [cbe::delegate::CreateRoleDelegate](#) that is passed into asynchronous version of method createRole()

12.75.2.3 JoinDelegatePtr

```
using cbe::Group::JoinDelegatePtr = delegate::group::JoinDelegatePtr
```

Pointer to [cbe::delegate::group::JoinDelegate](#) that is passed into asynchronous version of method join()

12.75.2.4 LeaveDelegatePtr

```
using cbe::Group::LeaveDelegatePtr = delegate::LeaveDelegatePtr
```

Pointer to [cbe::delegate::LeaveDelegate](#) that is passed into asynchronous version of method [leave\(\)](#)

12.75.2.5 ListBannedMembersDelegatePtr

```
using cbe::Group::ListBannedMembersDelegatePtr = delegate::ListMembersDelegatePtr
```

Pointer to [cbe::delegate::ListMembersDelegate](#) that is passed into asynchronous version of method [listBannedMembers\(\)](#)

12.75.2.6 ListMembersDelegatePtr

```
using cbe::Group::ListMembersDelegatePtr = delegate::ListMembersDelegatePtr
```

Pointer to [cbe::delegate::ListMembersDelegate](#) that is passed into asynchronous version of method [listMembers\(\)](#)

12.75.2.7 ListRolesDelegatePtr

```
using cbe::Group::ListRolesDelegatePtr = delegate::ListRolesDelegatePtr
```

Pointer to [cbe::delegate::ListRolesDelegate](#) that is passed into asynchronous version of method [listRoles\(\)](#)

12.75.2.8 RemoveDelegatePtr

```
using cbe::Group::RemoveDelegatePtr = delegate::group::RemoveDelegatePtr
```

Pointer to [cbe::delegate::RemoveDelegate](#) that is passed into asynchronous version of method [remove\(\)](#)

12.75.2.9 RemoveRoleDelegatePtr

```
using cbe::Group::RemoveRoleDelegatePtr = delegate::RemoveRoleDelegatePtr
```

Pointer to [cbe::delegate::RemoveRoleDelegate](#) that is passed into asynchronous version of method [removeRole\(\)](#)

12.75.2.10 RenameDelegatePtr

```
using cbe::Group::RenameDelegatePtr = delegate::group::RenameDelegatePtr
```

Pointer to [cbe::delegate::RenameDelegate](#) that is passed into asynchronous version of method [rename\(\)](#)

12.75.3 Constructor & Destructor Documentation

12.75.3.1 Group()

```
cbe::Group::Group (
    cbe::DefaultCtor )
```

Default constructor.

Construct a new object with the `DefaultCtor` to enable the `operator bool()` test

12.75.4 Member Function Documentation

12.75.4.1 createGroup()

```
cbe::Group cbe::Group::createGroup (
    std::string name,
    std::string memberAlias,
    cbe::Visibility visibility,
    CreateGroupDelegatePtr delegate )
```

Create a new [Group](#).

Note

Can only be used by Tenant admin/owners to create new groups.

Parameters

<i>name</i>	of the group
<i>memberAlias</i>	another popular name
<i>delegate</i>	Pointer to a delegate::CreateGroupDelegate instance that is implemented by the user.
<i>visibility</i>	default: public

Returns

[cbe::Group](#)

See also

[Types.h](#) for `cbe::visibility` enum `cbe::Visibility::PublicGroup` or `cbe::Visibility::Private`

12.75.4.2 createRole()

```
void cbe::Group::createRole (
    std::string name,
    CreateRoleDelegatePtr delegate )
```

Creates a new role in the group.

Parameters

<i>name</i>	The name of the role.
<i>delegate</i>	Pointer to a delegate::CreateRoleDelegate instance that is implemented by the user.

12.75.4.3 getVisibility()

```
cbe::Visibility cbe::Group::getVisibility ( ) const
```

Visibility of the [Group](#), Public is searchable and Private is not. In this early version you can create Private groups but the ability to invite has been blocked. (work in progress)

See also

[Types.h](#) for [cbe::PublishVisibility](#) enum

12.75.4.4 groupContainer()

```
cbe::Container cbe::Group::groupContainer ( ) const
```

Every group has a drive/container where resources can be shared with the members of the group. Works like a shared container.

12.75.4.5 id()

```
cbe::GroupId cbe::Group::id ( ) const
```

Returns the id number of the group.

12.75.4.6 join()

```
void cbe::Group::join (
    std::string alias,
    cbe::Visibility memberVisibility,
    std::string applicationComment,
    JoinDelegatePtr delegate )
```

Synchronous [exception] Ask to join a group. In this first version All members will be Public, meaning visible for other member inside the group. All groups will also be open so all join requests should be accepted directly.

Note

This is completely different from the query chain join.

Parameters

<i>alias</i>	name
<i>delegate</i>	Pointer to a delegate::group::JoinDelegate instance that is implemented by the user.
<i>memberVisibility</i>	default: public
<i>applicationComment</i>	message to the owner regarding the application to join

12.75.4.7 joined()

```
bool cbe::Group::joined ( ) const
```

Searched groups are obtained through the GroupQuery response. This list of groups have non-joined and joined groups. Already joined groups can be found on the `groupManager.groups()` and they are also cached. The joined bool is used to easily sort out groups in the GroupQuery.

12.75.4.8 leave()

```
void cbe::Group::leave (
    LeaveDelegatePtr delegate )
```

Leave group.

Parameters

<i>delegate</i>	Pointer to a delegate::LeaveDelegate instance that is implemented by the user.
-----------------	--

12.75.4.9 listBannedMembers()

```
void cbe::Group::listBannedMembers (
    ListBannedMembersDelegatePtr delegate )
```

Lists all banned former members, or users.

Parameters

<i>delegate</i>	Pointer to a delegate::ListMembersDelegate instance that is implemented by the user.
-----------------	--

12.75.4.10 listMembers()

```
void cbe::Group::listMembers (
    ListMembersDelegatePtr delegate )
```

List all members in the group.

The member list is then retrieved via the delegate callback function [cbe::delegate::ListMembersDelegate::onListMembersSuccess\(\)](#)

Parameters

<i>delegate</i>	Pointer to a delegate::ListMembersDelegate instance that is implemented by the user.
-----------------	--

12.75.4.11 listRoles()

```
void cbe::Group::listRoles (
    ListRolesDelegatePtr delegate )
```

Lists all roles in the group.

Parameters

<i>delegate</i>	Pointer to a delegate::ListRolesDelegate instance that is implemented by the user.
-----------------	--

12.75.4.12 name()

```
std::string cbe::Group::name ( ) const
```

Returns the name of the group.

12.75.4.13 operator bool()

```
cbe::Group::operator bool ( ) const [explicit]
```

Checks if the current instance is real.

An "unreal" instance implies typically a failed event.

Relies on the `Default` constructor [Group\(cbe::DefaultCtor\)](#)

Returns

`true` : is real
`false` : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

12.75.4.14 parentId()

`cbe::GroupId` `cbe::Group::parentId ( )` const
 Group's Parent id number.

12.75.4.15 remove()

`void` `cbe::Group::remove (`
 `RemoveDelegatePtr` `delegate` `)`
 Remove group.

Note

This is exclusive for Tenant group owners.

Parameters

<i>delegate</i>	Pointer to a <code>delegate::group::RemoveDelegate</code> instance that is implemented by the user.
-----------------	---

12.75.4.16 removeRole()

`void` `cbe::Group::removeRole (`
 `RoleId` `roleId`,
 `RemoveRoleDelegatePtr` `delegate` `)`

 @brief Removes a role in the group.

 @param roleId The id of the role to remove.

 @param delegate Pointer to a `delegate::RemoveRoleDelegate` instance
 that is implemented by the user.

12.75.4.17 rename()

`void` `cbe::Group::rename (`
 `std::string` `newName`,
 `RenameDelegatePtr` `delegate` `)`
 Rename the Group; group id does not change.

Parameters

<i>newName</i>	the new name for the group
<i>delegate</i>	Pointer to a <code>delegate::group::RenameDelegate</code> instance that is implemented by the user.

12.75.4.18 requests()

`Requests` `cbe::Group::requests ( )` const

Requests to join the group. To retrieve the list call `listJoinRequests(cbe::GroupDelegatePtr delegate)` on the group object.

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

- `include/cbe/Group.h`

12.76 cbe::GroupFilter Class Reference

To filter when searching a list of [Group](#).

```
#include <GroupFilter.h>
```

Public Member Functions

- **GroupFilter** (const [GroupFilter](#) &rh)
- **GroupFilter** ([GroupFilter](#) &&rh)
- [GroupFilter](#) & **operator=** (const [GroupFilter](#) &rh)
- [GroupFilter](#) & **operator=** ([GroupFilter](#) &&rh)
- `std::string` [getQuery](#) () const
- `std::string` [getFilter](#) () const
- `bool` [getAscending](#) () const
- `bool` [getDeleted](#) () const
- `bool` [getPublicFirst](#) () const
- `uint32_t` [getOffset](#) () const
- `uint32_t` [getCount](#) () const
- `std::string` [getOrder](#) () const
- `cbe::FilterOrder` [getGroupOrder](#) () const
- `cbe::GroupFilter` & [setQuery](#) (std::string)
- `cbe::GroupFilter` & [setFilter](#) (std::string)
- `cbe::GroupFilter` & [setAscending](#) (bool)
- `cbe::GroupFilter` & [setDeleted](#) (bool)
- `cbe::GroupFilter` & [setOffset](#) (uint32_t)
- `cbe::GroupFilter` & [setCount](#) (uint32_t)

Friends

- class **Group**
- class **GroupQueryResult**
- class **GroupManager**
- class **delegate::SearchGroupDelegate**
- `std::ostream` & [operator<<](#) (std::ostream &os, const [GroupFilter](#) &[GroupFilter](#))

12.76.1 Detailed Description

To filter when searching a list of [Group](#).

12.76.2 Member Function Documentation

12.76.2.1 getAscending()

```
bool cbe::GroupFilter::getAscending ( ) const
```

Returns the settings on how the data was sorted and displayed.

12.76.2.2 getCount()

```
uint32_t cbe::GroupFilter::getCount ( ) const
```

Returns the value of the count that was set for the query.

12.76.2.3 getDeleted()

```
bool cbe::GroupFilter::getDeleted ( ) const
```

If deleted objects in the bin are included.

12.76.2.4 getFilter()

```
std::string cbe::GroupFilter::getFilter ( ) const
```

the filter in question

12.76.2.5 getGroupOrder()

```
cbe::FilterOrder cbe::GroupFilter::getGroupOrder ( ) const
```

Returns the order the query was sorted check enum FilterOrder to see options for sorting.

12.76.2.6 getOffset()

```
uint32_t cbe::GroupFilter::getOffset ( ) const
```

Returns the offset that was used for the filter in the query.

12.76.2.7 getOrder()

```
std::string cbe::GroupFilter::getOrder ( ) const
```

get the filterorder as a string.

12.76.2.8 getPublicFirst()

```
bool cbe::GroupFilter::getPublicFirst ( ) const
```

If public groups should be sorted before private.

12.76.2.9 getQuery()

```
std::string cbe::GroupFilter::getQuery ( ) const
```

Returns the query string that was set on the filter. E.g., name:Abc*

12.76.2.10 setAscending()

```
cbe::GroupFilter& cbe::GroupFilter::setAscending (
    bool )
```

Sets the Order in which data should be displayed: Ascending meaning alphabetical

12.76.2.11 setCount()

```
cbe::GroupFilter& cbe::GroupFilter::setCount (
    uint32_t )
```

Set the Number of items you want to get from a container. So if a container has 50 items but you only want the 10 first in ascending order then set ascending to true and setCount to 10.

12.76.2.12 setDeleted()

```
cbe::GroupFilter& cbe::GroupFilter::setDeleted (
    bool )
```

If you want to see deleted groups set deleted to true.

12.76.2.13 setFilter()

```
cbe::GroupFilter& cbe::GroupFilter::setFilter (
    std::string )
```

Not fully tested but should be a regular expression.

12.76.2.14 setOffset()

```
cbe::GroupFilter& cbe::GroupFilter::setOffset (
    uint32_t )
```

Set the offset for paging, offset is the item offset where to start your query. i.e: There is already a query of the first 99 items and to get the rest you have setOffset to 100 to get the next set.

12.76.2.15 setQuery()

```
cbe::GroupFilter& cbe::GroupFilter::setQuery (
    std::string )
```

Set the query string, e.g: Name:* (would search for all objects with the metadata key of Name).

Note

If used with rootContainer id as the dataId, search in the whole account will be done.

12.76.3 Friends And Related Function Documentation

12.76.3.1 operator<<

```
std::ostream& operator<< (
    std::ostream & os,
    const GroupFilter & GroupFilter ) [friend]
```

Set the order of how data will be shown by the enum of FilterOrder ex: Titel first, published e.t.c

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

- include/cbe/GroupFilter.h

12.77 cbe::GroupManager Class Reference

For managing the groups.

```
#include <GroupManager.h>
```

Public Types

- using [ListGroupsDelegatePtr](#) = [delegate::ListGroupsDelegatePtr](#)
- using [SearchGroupsDelegatePtr](#) = [delegate::SearchGroupsDelegatePtr](#)

Public Member Functions

- void [listGroups](#) ([ListGroupsDelegatePtr](#) delegate)
- void [searchGroups](#) ([cbe::GroupFilter](#) filter, [SearchGroupsDelegatePtr](#) delegate, [cbe::GroupId](#) parentGroup↔ Id=0)
- [cbe::GroupId](#) [getTenantId](#) () const
- [GroupManager](#) ([cbe::DefaultCtor](#))
Default constructor.
- [operator bool](#) () const
Checks if the current instance is real.

Friends

- class [CloudBackend](#)

12.77.1 Detailed Description

For managing the groups.

12.77.2 Member Typedef Documentation

12.77.2.1 ListGroupsDelegatePtr

using `cbe::GroupManager::ListGroupsDelegatePtr = delegate::ListGroupsDelegatePtr`
 Pointer to `cbe::delegate::ListGroupsDelegate` that is passed into asynchronous version of method `listGroups()`

12.77.2.2 SearchGroupsDelegatePtr

using `cbe::GroupManager::SearchGroupsDelegatePtr = delegate::SearchGroupsDelegatePtr`
 Pointer to `SearchGroupsDelegate` that is passed into:

- `cbe::GroupManager::searchGroups()`

12.77.3 Constructor & Destructor Documentation

12.77.3.1 GroupManager()

```
cbe::GroupManager::GroupManager (
    cbe::DefaultCtor )
```

Default constructor.

Construct a new object with the `DefaultCtor` to enable the `operator bool()` test

12.77.4 Member Function Documentation

12.77.4.1 getTenantId()

```
cbe::GroupId cbe::GroupManager::getTenantId ( ) const
```

Pointer to `cbe::delegate::SearchGroupsDelegate` that is passed into asynchronous version of method `searchGroups()` Returns the tenant id of the Tenant user group that the user is in.

12.77.4.2 listGroups()

```
void cbe::GroupManager::listGroups (
    ListGroupsDelegatePtr delegate )
```

List all current joined groups.

Parameters

<i>delegate</i>	Pointer to a <code>delegate::ListGroupsDelegate</code> instance that is implemented by the user.
-----------------	--

12.77.4.3 operator bool()

```
cbe::GroupManager::operator bool ( ) const [explicit]
```

Checks if the current instance is real.

An "unreal" instance implies typically a failed event.

Relies on the Default constructor `GroupManager(cbe::DefaultCtor)`

Returns

`true` : is real

`false` : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

12.77.4.4 searchGroups()

```
void cbe::GroupManager::searchGroups (
    cbe::GroupFilter filter,
    SearchGroupsDelegatePtr delegate,
    cbe::GroupId parentGroupId = 0 )
```

Search for open public groups.

Parameters

<i>filter</i>	is a group filter to set search criteria for open public groups. Look att class GroupFilter for more information.
<i>delegate</i>	Pointer to a delegate::SearchGroupsDelegate instance that is implemented by the user.
<i>parent↵ GroupId</i>	is the id of the group to be searched within. If this is not set the tenant id will be used.

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

- include/cbe/GroupManager.h

12.78 cbe::GroupQueryResult Class Reference

Resultset of data retrieved in a search for [Group](#).

```
#include <GroupQueryResult.h>
```

Public Types

- using **GroupsSnapshot** = std::vector< [Group](#) >

Public Member Functions

- **GroupQueryResult** ([cbe::DefaultCtor](#))
- [cbe::GroupFilter](#) filter () const
- GroupsSnapshot [getGroupsSnapshot](#) () const
- uint64_t [GroupsLoaded](#) () const
- uint64_t [totalCount](#) () const
- bool [containsGroup](#) (uint64_t groupId)
- **operator bool** () const

Friends

- class **GroupManager**
- class **CloudBackend**
- std::ostream & **operator**<< (std::ostream &os, const [GroupQueryResult](#) &groupQuery)

12.78.1 Detailed Description

Resultset of data retrieved in a search for [Group](#).

12.78.2 Member Function Documentation

12.78.2.1 containsGroup()

```
bool cbe::GroupQueryResult::containsGroup (
    uint64_t groupId )
```

asking if the group with groupId was loaded in the query.

Parameters

<i>groupId</i>	the group id number in question
----------------	---------------------------------

12.78.2.2 filter()

```
cbe::GroupFilter cbe::GroupQueryResult::filter ( ) const
```

Returns a copy of the filter used for query.

12.78.2.3 getGroupsSnapshot()

```
GroupsSnapshot cbe::GroupQueryResult::getGroupsSnapshot ( ) const
```

Returns a copy of a vector containing the groups for the queryResult. The queryResult will update when new data comes in but the copy will not. If iterating, make sure to create a variable for a local copy.

Returns

vector<cbe::Group> containing the groups matching the query.

12.78.2.4 GroupsLoaded()

```
uint64_t cbe::GroupQueryResult::GroupsLoaded ( ) const
```

Number of groups loaded in the queryResult.

12.78.2.5 totalCount()

```
uint64_t cbe::GroupQueryResult::totalCount ( ) const
```

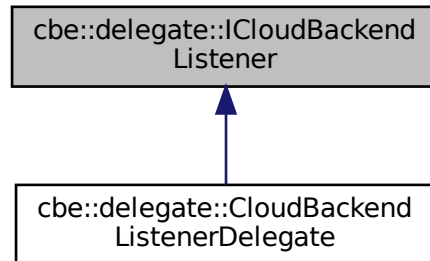
Total number of Groups in the cloud matching the query result. This may be more than loaded.

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

- include/cbe/GroupQueryResult.h

12.79 cbe::delegate::ICloudBackendListener Class Reference

Inheritance diagram for cbe::delegate::ICloudBackendListener:



Public Member Functions

- virtual void [onRemoteObjectAdded](#) (cbe::Object &&object)=0
- virtual void [onRemoteObjectMoved](#) (cbe::Object &&object)=0
- virtual void [onRemoteObjectRemoved](#) (cbe::ItemId objectId, std::string name)=0
- virtual void [onRemoteObjectRenamed](#) (cbe::Object &&object)=0
- virtual void [onRemoteContainerAdded](#) (cbe::Container &&container)=0
- virtual void [onRemoteContainerMoved](#) (cbe::Container &&container)=0
- virtual void [onRemoteContainerRemoved](#) (cbe::ItemId containerId, std::string name)=0
- virtual void [onRemoteContainerRenamed](#) (cbe::Container &&container)=0

12.79.1 Member Function Documentation

12.79.1.1 onRemoteContainerAdded()

```
virtual void cbe::delegate::ICloudBackendListener::onRemoteContainerAdded (
    cbe::Container && container ) [pure virtual]
```

Called upon successful Create.

Parameters

<i>container</i>	Instance of container that is being created.
------------------	--

Implemented in [cbe::delegate::CloudBackendListenerDelegate](#).

12.79.1.2 onRemoteContainerMoved()

```
virtual void cbe::delegate::ICloudBackendListener::onRemoteContainerMoved (
    cbe::Container && container ) [pure virtual]
```

Called upon successful Move.

Parameters

<i>container</i>	Instance of container that is being moved.
------------------	--

Implemented in [cbe::delegate::CloudBackendListenerDelegate](#).

12.79.1.3 onRemoteContainerRemoved()

```
virtual void cbe::delegate::ICloudBackendListener::onRemoteContainerRemoved (
    cbe::ItemId containerId,
    std::string name ) [pure virtual]
```

Called upon successful Remove.

Parameters

<i>containerId</i>	Instance of the container that is being Removed.
<i>name</i>	Name of the container that is being removed.

Implemented in [cbe::delegate::CloudBackendListenerDelegate](#).

12.79.1.4 onRemoteContainerRenamed()

```
virtual void cbe::delegate::ICloudBackendListener::onRemoteContainerRenamed (
    cbe::Container && container ) [pure virtual]
```

Called upon successful Rename.

Parameters

<i>container</i>	Instance of container that is being Renamed.
------------------	--

Implemented in [cbe::delegate::CloudBackendListenerDelegate](#).

12.79.1.5 onRemoteObjectAdded()

```
virtual void cbe::delegate::ICloudBackendListener::onRemoteObjectAdded (
    cbe::Object && object ) [pure virtual]
```

Called upon successful create of an [object](#).

Parameters

<i>object</i>	Instance of object that is being created.
---------------	---

Implemented in [cbe::delegate::CloudBackendListenerDelegate](#).

12.79.1.6 onRemoteObjectMoved()

```
virtual void cbe::delegate::ICloudBackendListener::onRemoteObjectMoved (
    cbe::Object && object ) [pure virtual]
```

Called upon successful move of an [object](#).

Parameters

<i>object</i>	Instance of object that is being moved.
---------------	---

Implemented in [cbe::delegate::CloudBackendListenerDelegate](#).

12.79.1.7 onRemoteObjectRemoved()

```
virtual void cbe::delegate::ICloudBackendListener::onRemoteObjectRemoved (
    cbe::ItemId objectId,
    std::string name ) [pure virtual]
```

Called upon successful removal of an [object](#).

Parameters

<i>objectId</i>	Instance of the object that is being Removed.
<i>name</i>	Name of the object that is being Removed.

Implemented in [cbe::delegate::CloudBackendListenerDelegate](#).

12.79.1.8 onRemoteObjectRenamed()

```
virtual void cbe::delegate::ICloudBackendListener::onRemoteObjectRenamed (
    cbe::Object && object ) [pure virtual]
```

Called upon successful renaming of an [object](#).

Parameters

<i>object</i>	Instance of object that is being renamed.
---------------	---

Implemented in [cbe::delegate::CloudBackendListenerDelegate](#).

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

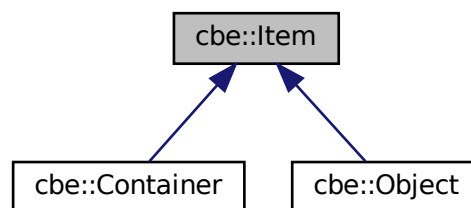
- include/cbe/delegate/ICloudBackendListener.h

12.80 cbe::Item Class Reference

A set made up of [Container](#) and [Object](#).

```
#include <Item.h>
```

Inheritance diagram for cbe::Item:



Public Member Functions

- [cbe::ShareIds](#) getShareIds () const
- [cbe::ShareId](#) getShareFromUserId ([cbe::UserId](#) userId)
- [cbe::UserId](#) getUserFromShareId ([cbe::UserId](#) shareId)
- std::string [aclTag](#) () const

- `std::string description () const`
- `cbe::ItemId id () const`
- `cbe::ContainerId parentId () const`
- `cbe::ContainerId oldParentId () const`
- `std::string name () const`
- `std::string path () const`
- `cbe::UserId ownerId () const`
- `cbe::ContainerId driveId () const`
- `std::string username () const`
- `bool idLoaded () const`
- `bool dataLoaded () const`
- `cbe::Date created () const`
- `cbe::Date updated () const`
- `cbe::Date deleted () const`
- `cbe::ItemType type () const`
- `bool hasPublished () const`
- `Publish getPublished () const`
- `bool hasSubscribe () const`
- `Subscribe getSubscribe () const`
- `bool operator< (const cbe::Item &rh) const`
- `AcIMap aclMap () const`
- `operator bool () const`

Checks if the inherited instance is real.

Protected Member Functions

- `Item (std::shared_ptr< Impl > pImpl)`
- `template<class ImplT >
ImplT & castImpl () const`

Friends

- `class CloudBackend`
- `class Container`
- `class Object`
- `class PublishManager`
- `class QueryResult`
- `class SubscribeManager`
- `std::ostream & operator<< (std::ostream &os, const Item &item)`

12.80.1 Detailed Description

A set made up of [Container](#) and [Object](#).

Class for an [Item](#). This class is the base class of Objects and Containers.

12.80.2 Member Function Documentation

12.80.2.1 aclMap()

`AcIMap cbe::Item::aclMap ( ) const`

Retrieve the map of the permissions the users and groups for this item.

12.80.2.2 `aclTag()`

`std::string cbe::Item::aclTag ( ) const`
the ACL of the item

12.80.2.3 `created()`

`cbe::Date cbe::Item::created ( ) const`
Returns the creation date in Unix epoch time.

12.80.2.4 `dataLoaded()`

`bool cbe::Item::dataLoaded ( ) const`
if data was loaded

12.80.2.5 `deleted()`

`cbe::Date cbe::Item::deleted ( ) const`
Returns the deleted date, if applicable (i.e. is in the bin), in Unix epoch time

12.80.2.6 `description()`

`std::string cbe::Item::description ( ) const`
A item text description where applicable

12.80.2.7 `driveId()`

`cbe::ContainerId cbe::Item::driveId ( ) const`
Returns which drive number the container resides on.

12.80.2.8 `getPublished()`

`Publish cbe::Item::getPublished ( ) const`
Gets the publish information

See also

[`hasPublished\(\)`](#)

12.80.2.9 `getShareFromUserId()`

`cbe::ShareId cbe::Item::getShareFromUserId (`
`cbe::UserId userId )`

Get the shareId by reference of userId

12.80.2.10 `getShareIds()`

`cbe::ShareIds cbe::Item::getShareIds ( ) const`
a map of share ids for this item

12.80.2.11 `getSubscribe()`

`Subscribe cbe::Item::getSubscribe ( ) const`
Gets the subscribe information

See also

[`hasSubscribe\(\)`](#)

12.80.2.12 getUserFromShareId()

```
cbe::UserId cbe::Item::getUserFromShareId (
    cbe::UserId shareId )
```

Get the userId reference of shareId

12.80.2.13 hasPublished()

```
bool cbe::Item::hasPublished ( ) const
```

Checks if it has published info and if [getPublished\(\)](#) returns a valid object.

12.80.2.14 hasSubscribe()

```
bool cbe::Item::hasSubscribe ( ) const
```

Checks if it has subscribe info and if [getSubscribe\(\)](#) returns a valid object.

12.80.2.15 id()

```
cbe::ItemId cbe::Item::id ( ) const
```

Returns an Items id.

12.80.2.16 idLoaded()

```
bool cbe::Item::idLoaded ( ) const
```

if the query did get a resultset.

12.80.2.17 name()

```
std::string cbe::Item::name ( ) const
```

Returns the name of the item.

12.80.2.18 oldParentId()

```
cbe::ContainerId cbe::Item::oldParentId ( ) const
```

Returns the id of the Items old parent container id in cases where the item has just been moved.

12.80.2.19 operator bool()

```
cbe::Item::operator bool ( ) const [explicit]
```

Checks if the inherited instance is real.

An "unreal" instance implies typically a failed event.

Relies on the Default constructor [cbe::DefaultCtor](#)

Returns

 true : is real

 false : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

12.80.2.20 ownerId()

```
cbe::UserId cbe::Item::ownerId ( ) const
```

Returns the owner id number.

12.80.2.21 parentId()

```
cbe::ContainerId cbe::Item::parentId ( ) const
```

Returns the id of the Items parent container id.

12.80.2.22 path()

`std::string cbe::Item::path ( ) const`
 Returns the path if applicable.

12.80.2.23 type()

`cbe::ItemType cbe::Item::type ( ) const`
[Container](#) or [Object](#), see enum in [Types.h](#)

12.80.2.24 updated()

`cbe::Date cbe::Item::updated ( ) const`
 Returns the updated date/time in Unix epoch time

12.80.2.25 username()

`std::string cbe::Item::username ( ) const`
 Returns the username of the owner if applicable.
 The documentation for this class was generated from the following file:

- `include/cbe/Item.h`

12.81 cbe::delegate::ItemError Class Reference

`#include <ItemDelegateErrors.h>`

Classes

- struct [Error](#)

Public Member Functions

- virtual void [onItemError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

Friends

- `std::ostream & operator<< (std::ostream &os, const ItemError::Error &error)`

12.81.1 Detailed Description

Entity class containing the information passed into method `onItemError()` in cbe interface

12.81.2 Member Function Documentation**12.81.2.1 onItemError()**

`virtual void cbe::delegate::ItemError::onItemError (`
 [Error](#) && error,
 [cbe::util::Context](#) && context) [pure virtual]

Called upon failed service calls like:

- move
- remove
- rename

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

- `include/cbe/delegate/ItemDelegateErrors.h`

12.82 cbe::delegate::group::JoinDelegate Class Reference

```
#include <JoinDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Group](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onJoinSuccess](#) ([cbe::Group](#) &&group)=0
- virtual void [onJoinError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.82.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Group::join](#)

Note

This is different from the query chain join.

12.82.2 Member Function Documentation

12.82.2.1 onJoinError()

```
virtual void cbe::delegate::group::JoinDelegate::onJoinError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.82.2.2 onJoinSuccess()

```
virtual void cbe::delegate::group::JoinDelegate::onJoinSuccess (
    cbe::Group && group ) [pure virtual]
```

Called upon successful Join.

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

- include/cbe/delegate/group/JoinDelegate.h

12.83 cbe::delegate::JoinDelegate Class Reference

```
#include <JoinDelegate.h>
```

Public Types

- using **Success** = [cbe::QueryResult](#)
- using **JoinError** = [QueryError](#)
- using **Error** = [JoinError](#)
- using [ErrorInfo](#) = [QueryJoinDelegate::ErrorInfo](#)

Public Member Functions

- virtual void [onJoinSuccess](#) ([cbe::QueryResult](#) &&queryResult)=0
- virtual void [onJoinError](#) ([JoinError](#) &&error, [cbe::util::Context](#) &&context)=0

12.83.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::QueryChain::join\(\)](#)

Note

This is different from the group join.

12.83.2 Member Typedef Documentation

12.83.2.1 ErrorInfo

```
using cbe::delegate::JoinDelegate::ErrorInfo = QueryJoinDelegate::ErrorInfo
```

Contains all information about a failed join.

12.83.3 Member Function Documentation

12.83.3.1 onJoinError()

```
virtual void cbe::delegate::JoinDelegate::onJoinError (  
    JoinError && error,  
    cbe::util::Context && context ) [pure virtual]
```

Called upon a failed join() call.

Parameters

<i>error</i>	Error information passed from CloudBackend SDK.
<i>context</i>	Additional context information about the original service call that has failed.

12.83.3.2 onJoinSuccess()

```
virtual void cbe::delegate::JoinDelegate::onJoinSuccess (  
    cbe::QueryResult && queryResult ) [pure virtual]
```

Called upon successful query.

Parameters

<i>queryResult</i>	Instance of a QueryResult containing the result set.
--------------------	--

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

- include/cbe/delegate/JoinDelegate.h

12.84 cbe::delegate::JoinError Class Reference

```
#include <ItemDelegateErrors.h>
```

Classes

- class [Error](#)

Public Member Functions

- virtual void [onJoinError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

Friends

- std::ostream & **operator**<< (std::ostream &os, const [JoinError::Error](#) &error)

12.84.1 Detailed Description

Entity class containing the information passed into method **onJoinError()** in cbe interface

12.84.2 Member Function Documentation

12.84.2.1 onJoinError()

```
virtual void cbe::delegate::JoinError::onJoinError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called upon a failed join() call.

Parameters

<i>error</i>	Error information passed from CloudBackend SDK.
<i>context</i>	Additional context information about the original service call that has failed.

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

- include/cbe/delegate/ItemDelegateErrors.h

12.85 cbe::delegate::KickDelegate Class Reference

```
#include <KickDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [KickSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onKickSuccess](#) (std::string &&memberName, [cbe::MemberId](#) memberId)=0
- virtual void [onKickError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.85.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Member::kick](#)

12.85.2 Member Function Documentation

12.85.2.1 onKickError()

```
virtual void cbe::delegate::KickDelegate::onKickError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.85.2.2 onKickSuccess()

```
virtual void cbe::delegate::KickDelegate::onKickSuccess (
    std::string && memberName,
    cbe::MemberId memberId ) [pure virtual]
```

Called upon successful ban.

Parameters

<i>memberName</i>	Name of the member being kicked.
<i>memberId</i>	Id of the member being kicked.

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

- include/cbe/delegate/KickDelegate.h

12.86 cbe::delegate::KickSuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::KickDelegate::onKickSuccess](#).

```
#include <KickDelegate.h>
```

Public Member Functions

- **KickSuccess** ([cbe::DefaultCtor](#))
- **KickSuccess** (std::string &&memberName, [cbe::MemberId](#) memberId)

Public Attributes

- std::string **memberName** {}
- [cbe::MemberId](#) **memberId** {}

12.86.1 Detailed Description

Convenience type that bundles all parameters passed to method [cbe::delegate::KickDelegate::onKickSuccess](#).

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

- include/cbe/delegate/KickDelegate.h

12.87 cbe::delegate::LeaveDelegate Class Reference

```
#include <LeaveDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [LeaveSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onLeaveSuccess](#) (std::string &&memberName, [cbe::MemberId](#) memberId)=0
- virtual void [onLeaveError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.87.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Group::leave](#)

12.87.2 Member Function Documentation

12.87.2.1 onLeaveError()

```
virtual void cbe::delegate::LeaveDelegate::onLeaveError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.87.2.2 onLeaveSuccess()

```
virtual void cbe::delegate::LeaveDelegate::onLeaveSuccess (
    std::string && memberName,
    cbe::MemberId memberId ) [pure virtual]
```

Called upon successful leave.

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

- include/cbe/delegate/LeaveDelegate.h

12.88 cbe::delegate::LeaveSuccess Class Reference

Public Member Functions

- **LeaveSuccess** ([cbe::DefaultCtor](#))
- **LeaveSuccess** (std::string &&memberName, [cbe::MemberId](#) memberId)
- **operator bool** () const
Checks if current instance is valid.

Public Attributes

- std::string **memberName** {}
- [cbe::MemberId](#) **memberId** {}

12.88.1 Member Function Documentation

12.88.1.1 operator bool()

`cbe::delegate::LeaveSuccess::operator bool ( ) const [explicit]`
Checks if current instance is valid.

Returns

`true`: is valid

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

- `include/cbe/delegate/LeaveDelegate.h`

12.89 cbe::delegate::ListGroupDelegate Class Reference

```
#include <ListGroupDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Groups** = `std::vector< cbe::Group >`
- using **Success** = `Groups`
- using **Error** = `delegate::Error`

Public Member Functions

- virtual void [onListGroupSuccess](#) (`Groups &&groups`)=0
- virtual void [onListGroupError](#) (`delegate::Error &&error, cbe::util::Context &&context`)=0

12.89.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::GroupManager::listGroups](#)

12.89.2 Member Function Documentation

12.89.2.1 onListGroupError()

```
virtual void cbe::delegate::ListGroupDelegate::onListGroupError (
    delegate::Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.89.2.2 onListGroupSuccess()

```
virtual void cbe::delegate::ListGroupDelegate::onListGroupSuccess (
    Groups && groups ) [pure virtual]
```

Called upon successful listGroup

Parameters

<i>groups</i>	Ref to vector of cbe::Group holding the joined groups.
---------------	--

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

- `include/cbe/delegate/ListGroupsDelegate.h`

12.90 cbe::delegate::ListMembersDelegate Class Reference

```
#include <ListMembersDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Members** = `std::vector< cbe::Member >`
- using **Success** = `Members`
- using **Error** = `delegate::Error`

Public Member Functions

- virtual void [onListMembersSuccess](#) (`Members &&members`)=0
- virtual void [onListMembersError](#) (`Error &&error, cbe::util::Context &&context`)=0

12.90.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Group::listMembers](#)

12.90.2 Member Function Documentation

12.90.2.1 onListMembersError()

```
virtual void cbe::delegate::ListMembersDelegate::onListMembersError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.90.2.2 onListMembersSuccess()

```
virtual void cbe::delegate::ListMembersDelegate::onListMembersSuccess (
    Members && members ) [pure virtual]
```

Called upon successful ListMembers.

Parameters

<i>members</i>	Vector of cbe::Member holding the members of a group.
----------------	---

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

- `include/cbe/delegate/ListMembersDelegate.h`

12.91 cbe::delegate::ListRolesDelegate Class Reference

```
#include <ListRolesDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Roles** = std::vector< [cbe::Role](#) >
- using **Success** = Roles
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onListRolesSuccess](#) (Roles &&roles)=0
- virtual void [onListRolesError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.91.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Group::ListRoles](#)

12.91.2 Member Function Documentation

12.91.2.1 onListRolesError()

```
virtual void cbe::delegate::ListRolesDelegate::onListRolesError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.91.2.2 onListRolesSuccess()

```
virtual void cbe::delegate::ListRolesDelegate::onListRolesSuccess (
    Roles && roles ) [pure virtual]
```

Called upon successful ListRoles.

Parameters

<i>roles</i>	A unordered map with role id as key and role name as value of a group.
--------------	--

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

- [include/cbe/delegate/ListRolesDelegate.h](#)

12.92 cbe::delegate::ListSharesDelegate Class Reference

```
#include <ListSharesDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::QueryResult](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onListSharesSuccess](#) ([cbe::QueryResult](#) &&qResult)=0
- virtual void [onListSharesError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.92.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::ShareManager::listMyShares](#)

12.92.2 Member Function Documentation

12.92.2.1 onListSharesError()

```
virtual void cbe::delegate::ListSharesDelegate::onListSharesError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.92.2.2 onListSharesSuccess()

```
virtual void cbe::delegate::ListSharesDelegate::onListSharesSuccess (
    cbe::QueryResult && qResult ) [pure virtual]
```

Called upon successful Share.

Parameters

<i>qResult</i>	Instance of cbe::QueryResult containing list of shares.
----------------	---

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

- include/cbe/delegate/ListSharesDelegate.h

12.93 cbe::delegate::LoadError Class Reference

```
#include <ItemDelegateErrors.h>
```

Classes

- class [Error](#)

Public Member Functions

- virtual void [onLoadError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

Friends

- std::ostream & **operator**<< (std::ostream &os, const [LoadError::Error](#) &error)

12.93.1 Detailed Description

Entity class containing the information passed into method **onLoadError()** in cbe interface

12.93.2 Member Function Documentation

12.93.2.1 onLoadError()

```
virtual void cbe::delegate::LoadError::onLoadError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called upon a failed query() or join() call.

Parameters

<i>error</i>	Error information passed from CloudBackend SDK.
<i>context</i>	Additional context information about the original service call that has failed.

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

- include/cbe/delegate/ItemDelegateErrors.h

12.94 cbe::delegate::LoginDelegate Class Reference

```
#include <LoginDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Error** = [cbe::delegate::Error](#)
- using **Success** = [CloudBackend](#)

Public Member Functions

- virtual void [onLoginSuccess](#) ([CloudBackend](#) &&cloudBackend)=0
- virtual void [onLoginError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.94.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::CloudBackend::login](#)(const std::string&,const std::string&,const std::string&,cbe::delegate::LoginDelegatePtr)
See [Example of asynchronous login](#)

12.94.2 Member Typedef Documentation

12.94.2.1 Success

```
using cbe::delegate::LoginDelegate::Success = CloudBackend
```

Type of object delivered on success. I.e., the object delivered through success callback [onLoginSuccess\(CloudBackend&&\)](#).

12.94.3 Member Function Documentation

12.94.3.1 onLogInError()

```
virtual void cbe::delegate::LogInDelegate::onLogInError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called upon failed log in.

Parameters

<i>error</i>	Error information passed from CloudBackend SDK.
<i>context</i>	Additional context information about the original service call that has failed.

12.94.3.2 onLogInSuccess()

```
virtual void cbe::delegate::LogInDelegate::onLogInSuccess (
    CloudBackend && cloudBackend ) [pure virtual]
```

Called upon successful log in.

Parameters

<i>cloudBackend</i>	Instance of a CloudBackend holding the session.
---------------------	---

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

- include/cbe/delegate/LogInDelegate.h

12.95 cbe::MatchesID< IdT > Class Template Reference

Search lists with ID.

```
#include <Utility.h>
```

Public Member Functions

- **MatchesID** (const IdT &id)
- `template<class ItemT >`
bool **operator()** (const ItemT &item) const

12.95.1 Detailed Description

```
template<typename IdT>
class cbe::MatchesID< IdT >
```

Search lists with ID.

Example:

```
bool alreadyExists = std::find_if(vector.begin(), vector.end(),
    cbe::MatchesID<uint64_t>(id)) != vector.end();
```

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

- include/cbe/Utility.h

12.96 cbe::Member Class Reference

A of member of a [Group](#).

```
#include <Member.h>
```

Public Types

- using `KickDelegatePtr` = `delegate::KickDelegatePtr`
- using `BanDelegatePtr` = `delegate::BanDelegatePtr`
- using `UnBanDelegatePtr` = `delegate::UnBanDelegatePtr`

Public Member Functions

- void `kick` (std::string kickComment, `KickDelegatePtr` delegate)
- void `ban` (std::string banComment, `BanDelegatePtr` delegate)
- void `unBan` (`UnBanDelegatePtr` delegate)
- `cbe::MemberId` `memberId` () const
- std::string `name` () const
- `cbe::Visibility` `visibility` () const
- `cbe::GroupId` `groupId` () const
- `cbe::MemberBanInfo` `getMemberBanInfo` ()
Gets the `member` ban info map.
- `Member` (`cbe::DefaultCtor`)
Default constructor.
- `operator bool` () const
Checks if the current instance is real.

Friends

- class `Group`
- class `Role`

12.96.1 Detailed Description

A of member of a `Group`.

12.96.2 Member Typedef Documentation

12.96.2.1 BanDelegatePtr

```
using cbe::Member::BanDelegatePtr = delegate::BanDelegatePtr
```

Pointer to `cbe::delegate::KickDelegate` that is passed into asynchronous version of method `kick()` Pointer to BanDelegate that is passed into:

- `cbe::Member::ban`(std::string,BanDelegatePtr)

12.96.2.2 KickDelegatePtr

```
using cbe::Member::KickDelegatePtr = delegate::KickDelegatePtr
```

Pointer to `KickDelegate` that is passed into:

- `cbe::Member::kick`(std::string,KickDelegatePtr)

12.96.2.3 UnBanDelegatePtr

```
using cbe::Member::UnBanDelegatePtr = delegate::UnBanDelegatePtr
```

Pointer to `cbe::delegate::BanDelegate` that is passed into asynchronous version of method `ban()` Pointer to UnBanDelegate that is passed into:

- `cbe::Member::unBan`(UnBanDelegatePtr)

12.96.3 Constructor & Destructor Documentation

12.96.3.1 Member()

```
cbe::Member::Member (
    cbe::DefaultCtor )
```

Default constructor.

Construct a new object with the `DefaultCtor` to enable the `operator bool()` test

12.96.4 Member Function Documentation

12.96.4.1 ban()

```
void cbe::Member::ban (
    std::string banComment,
    BanDelegatePtr delegate )
```

Ban or evict a member from a group

Parameters

<i>banComment</i>	Free text describing the reason for this decision
<i>delegate</i>	Shared pointer to the class in which you implement <code>cbe::delegate::BanDelegate</code> to receive the callback upon completion of this request.

12.96.4.2 groupId()

```
cbe::GroupId cbe::Member::groupId ( ) const
```

the group id number

12.96.4.3 kick()

```
void cbe::Member::kick (
    std::string kickComment,
    KickDelegatePtr delegate )
```

Kick out a member from a group

Parameters

<i>kickComment</i>	free text describing the reason for this decision
<i>delegate</i>	a shared pointer to the class in which you implement <code>cbe::delegate::KickDelegate</code> to receive the callback on server completion.

12.96.4.4 memberId()

```
cbe::MemberId cbe::Member::memberId ( ) const
```

Pointer to `cbe::delegate::UnBanDelegate` that is passed into asynchronous version of method `unBan()` the member id number

12.96.4.5 name()

```
std::string cbe::Member::name ( ) const
```

Returns the member name.

12.96.4.6 operator bool()

```
cbe::Member::operator bool ( ) const [explicit]
```

Checks if the current instance is real.

An "unreal" instance implies typically a failed event.

Relies on the Default constructor [Member\(cbe::DefaultCtor\)](#)

Returns

`true` : is real

`false` : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

12.96.4.7 unBan()

```
void cbe::Member::unBan (
    UnBanDelegatePtr delegate )
```

Revoke a ban of a member from a group

Parameters

<i>delegate</i>	a shared pointer to the class in which you implement cbe::delegate::UnBanDelegate to receive the callback on server completion.
-----------------	---

12.96.4.8 visibility()

```
cbe::Visibility cbe::Member::visibility ( ) const
```

see the enum `Visibility` in [Types.h](#)

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

- `include/cbe/Member.h`

12.97 cbe::delegate::container::MoveDelegate Class Reference

```
#include <MoveDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Container](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onMoveSuccess](#) ([cbe::Container](#) &&container)=0
- virtual void [onMoveError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.97.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Container::move](#)

12.97.2 Member Function Documentation

12.97.2.1 onMoveError()

```
virtual void cbe::delegate::container::MoveDelegate::onMoveError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.97.2.2 onMoveSuccess()

```
virtual void cbe::delegate::container::MoveDelegate::onMoveSuccess (
    cbe::Container && container ) [pure virtual]
```

Called upon successful Move.

Parameters

<i>container</i>	Instance of container that is being moved.
------------------	--

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

- include/cbe/delegate/container/MoveDelegate.h

12.98 cbe::delegate::object::MoveDelegate Class Reference

```
#include <MoveDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Object](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onMoveSuccess](#) ([cbe::Object](#) &&object)=0
- virtual void [onMoveError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.98.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Object::move](#)

12.98.2 Member Function Documentation

12.98.2.1 onMoveError()

```
virtual void cbe::delegate::object::MoveDelegate::onMoveError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.98.2.2 onMoveSuccess()

```
virtual void cbe::delegate::object::MoveDelegate::onMoveSuccess (
    cbe::Object && object ) [pure virtual]
```

Called upon successful Move.

Parameters

<i>object</i>	Instance of object that is being moved.
---------------	---

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

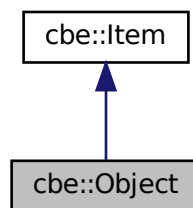
- include/cbe/delegate/object/MoveDelegate.h

12.99 cbe::Object Class Reference

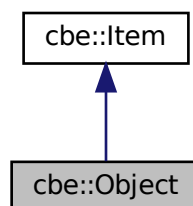
Holder of a set of data, can represent a table row.

```
#include <Object.h>
```

Inheritance diagram for cbe::Object:



Collaboration diagram for cbe::Object:



Public Types

- using [MoveDelegatePtr](#) = [delegate::object::MoveDelegatePtr](#)
- using [RenameDelegatePtr](#) = [delegate::object::RenameDelegatePtr](#)
- using [RemoveDelegatePtr](#) = [delegate::object::RemoveDelegatePtr](#)
- using [DownloadDelegatePtr](#) = [delegate::DownloadDelegatePtr](#)

- using `DownloadBinaryDelegatePtr` = `delegate::DownloadBinaryDelegatePtr`
- using `UploadDelegatePtr` = `delegate::UploadDelegatePtr`
- using `UpdateKeyValuesDelegatePtr` = `delegate::UpdateKeyValuesDelegatePtr`
- using `GetStreamsDelegatePtr` = `delegate::GetStreamsDelegatePtr`
- using `Streams` = `cbe::Streams`
Collection of `Stream` objects.
- using `GetAclDelegatePtr` = `delegate::AclDelegatePtr`
- using `SetAclDelegatePtr` = `delegate::AclDelegatePtr`
- using `ShareDelegatePtr` = `delegate::ShareDelegatePtr`
- using `UnShareDelegatePtr` = `delegate::UnShareDelegatePtr`
- using `PublishDelegatePtr` = `delegate::PublishDelegatePtr`
- using `UnPublishDelegatePtr` = `delegate::UnPublishDelegatePtr`
- using `UnSubscribeDelegatePtr` = `delegate::UnSubscribeDelegatePtr`

Public Member Functions

- void `move` (`cbe::ContainerId` dstId, `MoveDelegatePtr` delegate)
Relocates an object to a different container.
- void `rename` (const std::string &name, `RenameDelegatePtr` delegate)
Rename object.
- void `remove` (`RemoveDelegatePtr` delegate)
Remove the object from cloud and locally.
- void `download` (const std::string &path, `DownloadDelegatePtr` delegate)
Download the data of current object to the the local file system.
- void `download` (std::size_t &&sizeLimit, `DownloadBinaryDelegatePtr` delegate)
Download the binary data associated with current object.
- void `downloadStream` (const std::string &path, `cbe::Stream` stream, `DownloadDelegatePtr` delegate)
- void `uploadStream` (const std::string &filePath, `cbe::StreamId` streamId, `UploadDelegatePtr` delegate)
Upload a file for adding a new or replacing existing stream attached to this object.
- void `updateKeyValues` (`KeyValues` keyValues, `UpdateKeyValuesDelegatePtr` delegate)
Adds key/value pair data to the object.
- void `updateKeyValues` (`UpdateKeyValuesDelegatePtr` delegate)
Deletes all key/value pairs of data to the object.
- void `getStreams` (`GetStreamsDelegatePtr` delegate)
Downloads the streams meta data associated with current object to the SDK's cache.
- std::string `getMimeType` () const
Returns the mime type of the object.
- uint64_t `length` () const
Returns the binary length/size in bytes of current object.
- `cbe::object_t` `getObjectType` () const
Returns the `Object` type.
- `cbe::KeyValues` `keyValues` ()
Returns all the key/values for current object as a map.
- void `getAcl` (`GetAclDelegatePtr` delegate)
Returns the Access Control List for current `Object`.
- void `setAcl` (`cbe::AclMap` aclMap, `SetAclDelegatePtr` delegate)
Sets the Access Control List for current object.
- void `share` (`cbe::UserId` toUserGroup, std::string description, `ShareDelegatePtr` delegate)
Share current object to a user.
- void `unShare` (`cbe::ShareId` shareId, `UnShareDelegatePtr` delegate)
Unshare the object to a specific shareId created when sharing.

- void `publish` (`cbe::PublishAccess` security, `cbe::PublishVisibility` privacy, `std::string` description, `std::string` password, `PublishDelegatePtr` delegate)
Publishes current object to any user.
- void `unPublish` (`UnPublishDelegatePtr` delegate)
UnPublishes current object.
- void `unsubscribe` (`UnSubscribeDelegatePtr` delegate)
UnSubscribes from this object.
- `std::string` url ()
URL to current object.
- **Object** (`cbe::DefaultCtor`)

Friends

- class **CloudBackend**
- class **Container**

Additional Inherited Members

12.99.1 Detailed Description

Holder of a set of data, can represent a table row.

Note

The object name may not contain characters `<` `&` `:` `/`

Any key name must start with a letter or `_`

The following key names are reserved and should not be used: category, content, id, link and date

Key names are case sensitive, hence variations with uppercase are permitted.

12.99.2 Member Typedef Documentation

12.99.2.1 DownloadBinaryDelegatePtr

using `cbe::Object::DownloadBinaryDelegatePtr = delegate::DownloadBinaryDelegatePtr`

Pointer to `cbe::delegate::DownloadBinaryDelegate` that is passed into asynchronous version of method `download()`

12.99.2.2 DownloadDelegatePtr

using `cbe::Object::DownloadDelegatePtr = delegate::DownloadDelegatePtr`

Pointer to `cbe::delegate::DownloadDelegate` that is passed into asynchronous version of methods:

- `download(const std::string&, DownloadDelegatePtr)`
- `downloadStream(const std::string&, cbe::Stream, DownloadDelegatePtr)`

12.99.2.3 GetAclDelegatePtr

using `cbe::Object::GetAclDelegatePtr = delegate::AclDelegatePtr`

Pointer to `cbe::delegate::AclDelegate` that is passed into asynchronous version of method `getAcl()`

12.99.2.4 GetStreamsDelegatePtr

using `cbe::Object::GetStreamsDelegatePtr = delegate::GetStreamsDelegatePtr`

Pointer to `cbe::delegate::GetStreamsDelegate` that is passed into asynchronous version of method `getStream()`

12.99.2.5 MoveDelegatePtr

using `cbe::Object::MoveDelegatePtr = delegate::object::MoveDelegatePtr`

Pointer to `cbe::delegate::object::MoveDelegate` that is passed into asynchronous version of method `move()`

12.99.2.6 PublishDelegatePtr

using `cbe::Object::PublishDelegatePtr = delegate::PublishDelegatePtr`

Pointer to `cbe::delegate::PublishDelegate` that is passed into asynchronous version of method `publish()`

12.99.2.7 RemoveDelegatePtr

using `cbe::Object::RemoveDelegatePtr = delegate::object::RemoveDelegatePtr`

Pointer to `cbe::delegate::object::RenameDelegate` that is passed into asynchronous version of method `remove()`

12.99.2.8 RenameDelegatePtr

using `cbe::Object::RenameDelegatePtr = delegate::object::RenameDelegatePtr`

Pointer to `cbe::delegate::object::RenameDelegate` that is passed into asynchronous version of method `rename()`

12.99.2.9 SetAclDelegatePtr

using `cbe::Object::SetAclDelegatePtr = delegate::AclDelegatePtr`

Pointer to `cbe::delegate::AclDelegate` that is passed into asynchronous version of method `setAcl()`

12.99.2.10 ShareDelegatePtr

using `cbe::Object::ShareDelegatePtr = delegate::ShareDelegatePtr`

Pointer to `cbe::delegate::ShareDelegate` that is passed into asynchronous version of method `share()`

12.99.2.11 Streams

using `cbe::Object::Streams = cbe::Streams`

Collection of `Stream` objects.

See `cbe::Streams`

12.99.2.12 UnPublishDelegatePtr

using `cbe::Object::UnPublishDelegatePtr = delegate::UnPublishDelegatePtr`

Pointer to `cbe::delegate::UnPublishDelegate` that is passed into asynchronous version of method `unPublish()`

12.99.2.13 UnShareDelegatePtr

using `cbe::Object::UnShareDelegatePtr = delegate::UnShareDelegatePtr`

Pointer to `cbe::delegate::UnShareDelegate` that is passed into asynchronous version of method `unShare()`

12.99.2.14 UnSubscribeDelegatePtr

using `cbe::Object::UnSubscribeDelegatePtr = delegate::UnSubscribeDelegatePtr`

Pointer to `cbe::delegate::UnSubscribeDelegate` that is passed into asynchronous version of method `unSubscribe()`

12.99.2.15 UpdateKeyValuesDelegatePtr

using `cbe::Object::UpdateKeyValuesDelegatePtr = delegate::UpdateKeyValuesDelegatePtr`

Pointer to `cbe::delegate::UpdateKeyValuesDelegate` that is passed into asynchronous version of method `updateKeyValues()`

12.99.2.16 UploadDelegatePtr

using `cbe::Object::UploadDelegatePtr = delegate::UploadDelegatePtr`

Pointer to `cbe::delegate::UploadDelegate` that is passed into asynchronous version of method `uploadStream()`

12.99.3 Member Function Documentation

12.99.3.1 download() [1/2]

```
void cbe::Object::download (
    const std::string & path,
    DownloadDelegatePtr delegate )
```

Download the data of current object to the the local file system.

Asynchronous version of this service function.

The data will be contained in file, named after the name of current object (method `name()`), to the location given by parameter `path`.

Parameters

<i>path</i>	Folder location, on the local file system, of the file to be downloaded. This string must end with a slash ("/").
<i>delegate</i>	Pointer to a <code>delegate::DownloadDelegate</code> instance that is implemented by the user.

12.99.3.2 download() [2/2]

```
void cbe::Object::download (
    std::size_t && sizeLimit,
    DownloadBinaryDelegatePtr delegate )
```

Download the binary data associated with current object.

Asynchronous version of this service function.

The data, delivered as a BLOB (Binary Large Object), via parameter `data` in the the callback method `cbe::delegate::DownloadBinaryDelegate::onDownloadBinarySuccess()`.

Parameters

<i>delegate</i>	Pointer to a <code>delegate::DownloadBinaryDelegate</code> instance that is implemented by the user.
<i>sizeLimit</i>	Blocks anything larger than the size limit the user inputs. Prevents accidental downloads of too large objects on to the device.

12.99.3.3 downloadStream()

```
void cbe::Object::downloadStream (
    const std::string & path,
    cbe::Stream stream,
    DownloadDelegatePtr delegate )
```

Download a stream of an `Object` to local filesystem.

Asynchronous version of this service function.

Parameters

<i>path</i>	Folder location, on the local file system, of the file to be downloaded. This string must end with a slash ("/").
-------------	---

Parameters

<i>stream</i>	Get which stream you want by first calling <code>getStream()</code> and then choose which one to download.
<i>delegate</i>	Pointer to a delegate::DownloadDelegate instance that is implemented by the user.

Example

Async downloading the data attached to a [cbe::Object](#) via a [Stream](#) with **Object::downloadStream()**
Continuation of: [Async uploading data to a cbe::Object via a Stream with Object::uploadStream\(\)](#)

```

~~~
#include "cbe/delegate/DownloadDelegate.h"
~~~
struct MyDownloadDelegate : cbe::delegate::DownloadDelegate {
    std::mutex                mutex{};
    std::condition_variable    conditionVariable{};
    bool                      called{};
    Success success{};
    void onDownloadSuccess(cbe::Object&& object,
                          std::string path) final {
        {
            std::lock_guard<std::mutex> lock{mutex};
            success = Success{std::move(object), std::move(path)};
            called = true;
        }
        conditionVariable.notify_one();
    }
    void onDownloadError(cbe::delegate::TransferError&& transferError,
                        cbe::util::Context&& context) final {
        {
            std::lock_guard<std::mutex> lock{mutex};
            success = Success{};
            called = true;
        }
        conditionVariable.notify_one();
    }
    cbe::Object waitForRsp() {
        std::unique_lock<std::mutex> lock{mutex};
        conditionVariable.wait(lock, [this]{ return called; });
        // Reset called flag, so current delegate instance can be reused
        called = false;
        // If object is still default constructed, this implies a failed
        // cbe::Object::downloadStream() operation
        return success.object;
    }
}; // struct MyDownloadDelegate
~~~
std::shared_ptr<MyDownloadDelegate> myDownloadDelegate =
    std::make_shared<MyDownloadDelegate>();
constexpr const char* const downloadPath = "/tmp/download/";
for (const auto& stream : streams2) {
    const auto path = std::string{downloadPath} +
        std::to_string(stream.streamId);
    object.downloadStream(path, stream, myDownloadDelegate);
    if (!myDownloadDelegate->waitForRsp()) {
        // Failure, bail out
        ~~~
    }
}

```

12.99.3.4 getAcl()

```

void cbe::Object::getAcl (
    GetAclDelegatePtr delegate )

```

Returns the Access Control List for current [Object](#).

Asynchronous version of this service function.

Parameters

<i>delegate</i>	Pointer to a delegate::AclDelegate instance that is implemented by the user.
-----------------	--

12.99.3.5 getMimeType()

std::string cbe::Object::getMimeType () const

Returns the mime type of the object.

E.g., application/pdf, audio/wav, image/jpg, text/xml, video/mp4 etc.

12.99.3.6 getObjectType()

cbe::object_t cbe::Object::getObjectType () const

Returns the [Object](#) type.

See [cbe::ObjectType](#).

12.99.3.7 getStreams()

```
void cbe::Object::getStreams (
    GetStreamsDelegatePtr delegate )
```

Downloads the streams meta data associated with current object to the SDK's cache.

Asynchronous version of this service function.

The meta data is delivered as [cbe::Streams](#) via the delegate callback method [cbe::delegate::GetStreamsDelegate::onGetStreamsSuccess](#).

Further, the actual stream data are retrieved through method [downloadStream\(const std::string&,cbe::Stream,DownloadDelegatePtr\)](#).

Note

This method must be re-called if you upload more streams, see [uploadStream\(\)](#)

Parameters

<i>delegate</i>	Pointer to a delegate::GetStreamsDelegate instance that is implemented by the user.
-----------------	---

Example

Async retrieving the Streams attached to a [cbe::Object](#) with the [Object::getStreams\(\)](#)

Continuation of: [Async creation of a cbe::Object by using Container::upload\(\)](#)

```
~~~
#include "cbe/delegate/GetStreamsDelegate.h"
#include "cbe/util/Optional.h"
~~~
struct MyGetStreamsDelegate : cbe::delegate::GetStreamsDelegate {
    std::mutex mutex{};
    std::condition_variable conditionVariable{};
    bool called{};
    // Represents the result stream meta data as a cbe::Optional. Where an empty
    // Optional implies an error
    cbe::util::Optional<cbe::Streams> streams{};
    void onGetStreamsSuccess(cbe::Streams&& streams) final {
        {
            std::lock_guard<std::mutex> lock{mutex};
            this->streams = std::move(streams);
            called = true;
        }
        conditionVariable.notify_one();
    }
    void onGetStreamsError(cbe::delegate::Error&& error,
                          cbe::util::Context&& context) final {
        {
            std::lock_guard<std::mutex> lock{mutex};
            object = cbe::Object{ cbe::DefaultCtor{} };
            called = true;
        }
        conditionVariable.notify_one();
    }
    cbe::util::Optional<cbe::Streams> waitForRsp() {
        std::unique_lock<std::mutex> lock{mutex};
        conditionVariable.wait(lock, [this]{ return called; });
        // Reset called flag, so current delegate instance can be reused
        called = false;
        // If streams is still default constructed, this implies a failed
        // cbe::Object::getStreams() operation
    }
};
```

```

        return std::move(streams);
    }
}; // struct MyGetStreamsDelegate
~~~
std::shared_ptr<MyGetStreamsDelegate> getStreamsDelegate =
    std::make_shared<MyGetStreamsDelegate>();
object.getStreams(getStreamsDelegate);
// Simply make use of the Optional bool type conversion operator on the return
// value from the waitForRsp() method to determine the outcome of the
// getStreams() invocation above
cbe::util::Optional<cbe::Streams> streamsOpt1 =
    getStreamsDelegate->waitForRsp();

if (!streamsOpt1) {
    // Failure, bail out
    ~~~
}
// Now, streamsOpt1 contains the streams meta data, so move it to variable
// streams1
const cbe::Object::Streams streams1 = std::move(*streamsOpt1);
~~~

```

Continues at: [Async uploading data to a cbe::Object via a Stream with Object::uploadStream\(\)](#)

12.99.3.8 move()

```

void cbe::Object::move (
    cbe::ContainerId dstId,
    MoveDelegatePtr delegate )

```

Relocates an object to a different container.

Asynchronous version of this service function.

Parameters

<i>dstId</i>	Id of the destination container.
<i>delegate</i>	Pointer to a delegate::object::MoveDelegate instance that is implemented by the user.

12.99.3.9 publish()

```

void cbe::Object::publish (
    cbe::PublishAccess security,
    cbe::PublishVisibility privacy,
    std::string description,
    std::string password,
    PublishDelegatePtr delegate )

```

Publishes current object to any user.

Asynchronous version of this service function.

Can be revoked with [unPublish\(\)](#)

Parameters

<i>security</i>	A cbe::PublishAccess enum
<i>privacy</i>	A cbe::WebShareVisibility enum
<i>description</i>	Free text
<i>password</i>	Password
<i>delegate</i>	Pointer to a delegate::PublishDelegate instance that is implemented by the user.

12.99.3.10 remove()

```

void cbe::Object::remove (

```

```
RemoveDelegatePtr delegate )
```

Remove the object from cloud and locally.

Asynchronous version of this service function.

Parameters

<i>delegate</i>	Pointer to a delegate::object::RemoveDelegate instance that is implemented by the user.
-----------------	---

12.99.3.11 rename()

```
void cbe::Object::rename (
    const std::string & name,
    RenameDelegatePtr delegate )
```

Rename object.

Asynchronous version of this service function.

Parameters

<i>name</i>	string name of the object.
<i>delegate</i>	Pointer to a delegate::object::RenameDelegate instance that is implemented by the user.

12.99.3.12 setAcl()

```
void cbe::Object::setAcl (
    cbe::AclMap aclMap,
    SetAclDelegatePtr delegate )
```

Sets the Access Control List for current object.

Asynchronous version of this service function.

Parameters

<i>aclMap</i>	The desired permission for current object.
<i>delegate</i>	Pointer to a delegate::AclDelegate instance that is implemented by the user.

12.99.3.13 share()

```
void cbe::Object::share (
    cbe::UserId toUserGroup,
    std::string description,
    ShareDelegatePtr delegate )
```

Share current object to a user.

Asynchronous version of this service function.

Notifies the user that a share has occurred so that the user can check what permissions they have been given. Sharing gives read permissions as of right now but might change in the future.

Parameters

<i>toUserGroup</i>	Takes a user id or group id (lastly named is for the future) and share to.
<i>description</i>	Names the specific share between you and the user/group.
<i>delegate</i>	Pointer to a delegate::ShareDelegatePtr instance that is implemented by the user.

12.99.3.14 unPublish()

```
void cbe::Object::unPublish (
    UnPublishDelegatePtr delegate )
```

UnPublishes current object.

Asynchronous version of this service function.

Revokes previous [publish\(\)](#).

Parameters

<i>delegate</i>	Pointer to a delegate::UnPublishDelegate instance that is implemented by the user.
-----------------	--

12.99.3.15 unShare()

```
void cbe::Object::unShare (
    cbe::ShareId shareId,
    UnShareDelegatePtr delegate )
```

Unshare the object to a specific shareId created when sharing.

Asynchronous version of this service function.

Each share is unique between user/group and the one sharing. This is represented with a unique share id.

Parameters

<i>shareId</i>	The unique id for a share between the owner and other user/group.
<i>delegate</i>	Pointer to a delegate::UnShareDelegate instance that is implemented by the user.

12.99.3.16 unSubscribe()

```
void cbe::Object::unSubscribe (
    UnSubscribeDelegatePtr delegate )
```

UnSubscribes from this object.

Asynchronous version of this service function.

Revokes the subscription previously established with [cbe::SubscribeManager::subscribe\(\)](#)

Parameters

<i>delegate</i>	Pointer to a delegate::UnSubscribeDelegate instance that is implemented by the user.
-----------------	--

12.99.3.17 updateKeyValues() [1/2]

```
void cbe::Object::updateKeyValues (
    KeyValues keyValues,
    UpdateKeyValuesDelegatePtr delegate )
```

Adds key/value pair data to the object.

Asynchronous version of this service function.

Note

All existing key will be overwritten, new created.

Any key name must start with a letter or _

The following key names are reserved and should not be used: category, content, id, link and date

Key names are case sensitive, hence variations with uppercase are permitted.

Parameters

<i>keyValues</i>	Map of key/value pairs (metadata).
<i>delegate</i>	Pointer to a delegate::UpdateKeyValuesDelegate instance that is implemented by the user.

12.99.3.18 updateKeyValues() [2/2]

```
void cbe::Object::updateKeyValues (
    UpdateKeyValuesDelegatePtr delegate )
```

Deletes all key/value pairs of data to the object.

Same as [updateKeyValues\(KeyValues,UpdateKeyValuesDelegatePtr\)](#), but without the `keyValues` parameter.

Note

Any and all existing key/values will be erased.

12.99.3.19 uploadStream()

```
void cbe::Object::uploadStream (
    const std::string & filePath,
    cbe::StreamId streamId,
    UploadDelegatePtr delegate )
```

Upload a file for adding a new or replacing existing stream attached to this object.

Asynchronous version of this service function.

Requires that method [getStreams\(GetStreamsDelegatePtr\)](#) is called to identify all streams associated with current object.

Parameters

<i>filePath</i>	Fully qualified file name. I.e., the path, relative or absolute, including file name.
<i>streamId</i>	If the stream id already exists, it will be overwritten.
<i>delegate</i>	Pointer to a delegate::UploadDelegate instance that is implemented by the user.

Example

Async uploading data to a [cbe::Object](#) via a [Stream](#) with [Object::uploadStream\(\)](#)

Continuation of: [Async retrieving the Streams attached to a cbe::Object with the Object::getStreams\(\)](#)

```
~~~
#include "cbe/delegate/UploadDelegate.h"
~~~
#include <algorithm> // std::max_element
#include <vector>    // std::cbegin, std::cend
~~~
struct MyUploadDelegate : cbe::delegate::UploadDelegate {
    std::mutex                mutex{};
    std::condition_variable   conditionVariable{};
    bool                     called{};
    // Default construct, further see comment for object member in
    // MyUploadDelegate above
    cbe::Object object{ cbe::DefaultCtor() };
};
```

```

void onUploadSuccess(cbe::Object&& object) final {
    {
        std::lock_guard<std::mutex> lock{mutex};
        this->object = std::move(object);
        called = true;
    }
    conditionVariable.notify_one();
}
void onUploadError(cbe::delegate::TransferError&& transferError,
                  cbe::util::Context&& context) final {
    {
        std::lock_guard<std::mutex> lock{mutex};
        object = cbe::Object{ cbe::DefaultCtor{} };
        called = true;
    }
    conditionVariable.notify_one();
}
cbe::Object waitForRsp() {
    std::unique_lock<std::mutex> lock{mutex};
    conditionVariable.wait(lock, [this]{ return called; });
    // Reset called flag, so current delegate instance can be reused
    called = false;
    // If object is still default constructed, this implies a failed
    // cbe::Object::uploadStream() operation
    return object;
}
}; // struct MyUploadDelegate
~~~
constexpr const char* const myFile2Name = "myStream2";
const std::string      qualFile2Name = std::string(uploadPath) + myFile2Name;

std::ofstream ofs2(qualFile2Name);
ofs2 << "Line 21\n" << "Line 22\n" << "Line 23\n" << "Line 24\n"
    << std::flush;
ofs2.close();
const auto nextStreamId =
    std::max_element(std::cbegin(streams1), std::cend(streams1),
        [](const cbe::Stream& stream1,
           const cbe::Stream& stream2) {
            return stream1.streamId < stream2.streamId;
        })->streamId + 1;
std::shared_ptr<MyUploadDelegate> myUploadDelegate =
    std::make_shared<MyUploadDelegate>();
object.uploadStream(qualFile2Name, nextStreamId, myUploadDelegate);
// Use of the bool type conversion operator to detect success or failure
// as above
if (!myUploadDelegate->waitForRsp()) {
    // Failure, bail out
    ~~~
}
object.getStreams(getStreamsDelegate);
auto streamsOpt2 = getStreamsDelegate->waitForRsp();
if (!streamsOpt2) {
    // Failure, bail out
    ~~~
}
const auto streams2 = std::move(*streamsOpt2);
~~~

```

Continues at: [Async downloading the data attached to a cbe::Object via a Stream with Object::downloadStream\(\)](#)

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

- include/cbe/Object.h

12.100 cbe::util::Optional< T > Class Template Reference

Class template [Optional](#) manages an optional contained value - i.e., a value that is either present or absent.

#include <Optional.h>

Classes

- class [BadAccess](#)

Thrown by [value\(\)](#) method when accessing an object that does not contain a value.

Public Member Functions

- [Optional](#) ()=default

- Creates an empty *Optional*, i.e., no value present.
- **Optional** (const T &value)
Copy value ctor.
- **Optional** (T &&value) noexcept(std::is_nothrow_move_constructible< T >::value)
Move value ctor.
- **Optional** (const **Optional** &rh)
Copy ctor.
- **Optional** (**Optional** &&rh) noexcept(std::is_nothrow_move_constructible< T >::value)
Move ctor.
- **Optional** & **operator=** (const **Optional** &rh)
Copy assignment operator.
- **Optional** & **operator=** (**Optional** &&rh) noexcept
Move assignment operator.
- T & **value** ()
Value access non-const method.
- T & **operator*** ()
- T * **operator->** ()
- const T & **value** () const
Value access const method.
- const T & **operator*** () const
- const T * **operator->** () const
- bool **hasValue** () const noexcept
Checks whether a value is present.
- **operator bool** () const noexcept
Checks whether a value is present.
- void **reset** ()
Invalidates possible contained value. The contained value will be destructed.
- void **swap** (**Optional** &other) noexcept
Swaps the contents with those of other.
- template<typename... ArgTs>
void **emplace** (ArgTs &&... args)
Constructs the contained value in-place. If **this* already contains a value before the call, the contained value is destroyed by calling its destructor.

12.100.1 Detailed Description

```
template<typename T>
class cbe::util::Optional< T >
```

Class template **Optional** manages an optional contained value - i.e., a value that is either present or absent.

Template Parameters

<i>T</i>	Type of the contained value
----------	-----------------------------

12.100.2 Member Function Documentation

12.100.2.1 operator bool()

```
template<typename T >
cbe::util::Optional< T >::operator bool ( ) const [inline], [explicit], [noexcept]
```

Checks whether a value is present.

```
cbe::util::Optional<int> inquireInt(const char* prompt) {
    while (true) {
        std::cout << prompt << ": ";
        std::string answer;
        std::cin >> answer;
        try {
            return std::stoi(answer); // I.e., return cbe::util::Optional<int>{std::stoi(answer)}
        }
        catch (...) {
            // Invalid input
            return {}; // I.e., return cbe::util::Optional<int>{};
        }
    }
}

~~~
auto n = inquireInt("Please, give an integer value");
if (n) {
    std::cout << "Thanks for the number" << *n << "!\n";
} else {
    std::cout << "Not a number!\n";
}

~~~
```

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

- include/cbe/util/Optional.h

12.101 cbe::Publish Class Reference

Managing a published [Item](#).

```
#include <Publish.h>
```

Public Member Functions

- void [setTitle](#) (const std::string &title)
Set the Title of this publication.
- void [setSecurity](#) ([cbe::PublishAccess](#) security)
Set the Security of this publication.
- void [setPrivacy](#) ([cbe::PublishVisibility](#) privacy)
Set the Privacy of this publication.
- void [setPassword](#) (const std::string &password)
Set the Password of this publication.
- void [setDescription](#) (const std::string &description)
Set the Description of this publication.
- std::string [getTitle](#) () const
- [cbe::PublishAccess](#) [getSecurity](#) () const
- [cbe::PublishVisibility](#) [getPrivacy](#) () const
- std::string [getPassword](#) () const
- std::string [getDescription](#) () const
- [cbe::PublishId](#) [getPublishId](#) () const
- [Publish](#) ([cbe::DefaultCtor](#))
Default constructor.
- [operator bool](#) () const
Checks if the current instance is real.

Friends

- class [CloudBackend](#)
- class [Item](#)

12.101.1 Detailed Description

Managing a published [Item](#).

12.101.2 Constructor & Destructor Documentation

12.101.2.1 Publish()

```
cbe::Publish::Publish (
    cbe::DefaultCtor )
```

Default constructor.

Construct a new object with the `DefaultCtor` to enable the `operator bool()` test

12.101.3 Member Function Documentation

12.101.3.1 getDescription()

```
std::string cbe::Publish::getDescription ( ) const
```

Gets the description

12.101.3.2 getPassword()

```
std::string cbe::Publish::getPassword ( ) const
```

Checks if a password has been set.

Returns

"true" : if a password has been set

"" : empty string if not.

12.101.3.3 getPrivacy()

```
cbe::PublishVisibility cbe::Publish::getPrivacy ( ) const
```

Gets the privacy setting, see `cbe::PublishVisibility` enum

12.101.3.4 getPublishId()

```
cbe::PublishId cbe::Publish::getPublishId ( ) const
```

Gets the publish id number

12.101.3.5 getSecurity()

```
cbe::PublishAccess cbe::Publish::getSecurity ( ) const
```

Gets the security setting, see `cbe::PublishAccess` enum

12.101.3.6 getTitle()

```
std::string cbe::Publish::getTitle ( ) const
```

Gets the title

12.101.3.7 operator bool()

```
cbe::Publish::operator bool ( ) const [explicit]
```

Checks if the current instance is real.

An "unreal" instance implies typically a failed event.

Relies on the `Default` constructor `Publish(cbe::DefaultCtor)`

Returns

`true` : is real
`false` : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

12.101.3.8 setDescription()

```
void cbe::Publish::setDescription (
    const std::string & description )
```

Set the Description of this publication.

Parameters

<i>description</i>	free text
--------------------	-----------

12.101.3.9 setPassword()

```
void cbe::Publish::setPassword (
    const std::string & password )
```

Set the Password of this publication.

Parameters

<i>password</i>	text
-----------------	------

12.101.3.10 setPrivacy()

```
void cbe::Publish::setPrivacy (
    cbe::PublishVisibility privacy )
```

Set the Privacy of this publication.

Parameters

<i>privacy</i>	of type cbe::PublishVisibility enum
----------------	---

12.101.3.11 setSecurity()

```
void cbe::Publish::setSecurity (
    cbe::PublishAccess security )
```

Set the Security of this publication.

Parameters

<i>security</i>	of type cbe::PublishAccess enum
-----------------	---

12.101.3.12 setTitle()

```
void cbe::Publish::setTitle (
```

```
const std::string & title )
```

Set the Title of this publication.

Parameters

<i>title</i>	name
--------------	------

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

- include/cbe/Publish.h

12.102 cbe::delegate::PublishDelegate Class Reference

```
#include <PublishDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [PublishSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onPublishSuccess](#) ([cbe::Items](#) &&items)=0
- virtual void [onPublishError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.102.1 Detailed Description

Delegate class for the asynchronous version of methods:

- [cbe::Container::publish](#)
- [cbe::Object::publish](#)

12.102.2 Member Function Documentation

12.102.2.1 onPublishError()

```
virtual void cbe::delegate::PublishDelegate::onPublishError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.102.2.2 onPublishSuccess()

```
virtual void cbe::delegate::PublishDelegate::onPublishSuccess (
    cbe::Items && items ) [pure virtual]
```

Called upon successful publish.

Parameters

<i>items</i>	Instance of items that are being published.
--------------	---

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

- include/cbe/delegate/PublishDelegate.h

12.103 cbe::PublishManager Class Reference

Managing the list of web shares.

```
#include <PublishManager.h>
```

Public Member Functions

- [cbe::Items getPublished](#) () const
Lists publications that are shared with other [userid](#)s.
- [PublishManager](#) ([cbe::DefaultCtor](#))
Default constructor.
- [operator bool](#) () const
Checks if the current instance is real.

Friends

- class [CloudBackend](#)

12.103.1 Detailed Description

Managing the list of web shares.

12.103.2 Constructor & Destructor Documentation

12.103.2.1 PublishManager()

```
cbe::PublishManager::PublishManager (  
    cbe::DefaultCtor )
```

Default constructor.

Construct a new object with the `DefaultCtor` to enable the [operator bool\(\)](#) test

12.103.3 Member Function Documentation

12.103.3.1 getPublished()

```
cbe::Items cbe::PublishManager::getPublished ( ) const
```

Lists publications that are shared with other [userid](#)s.

Listing is done independently of where in the actual [container/subcontainer](#) tree the [items](#) are located.

(Publications are also known as web shares.)

Returns

The publications in terms of [items](#).

12.103.3.2 operator bool()

```
cbe::PublishManager::operator bool ( ) const [explicit]
```

Checks if the current instance is real.

An "unreal" instance implies typically a failed event.

Relies on the Default constructor [PublishManager\(cbe::DefaultCtor\)](#)

Returns

`true` : is real

`false` : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

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

- include/cbe/PublishManager.h

12.104 cbe::delegate::PublishSuccess Class Reference

Convenience type that bundles the parameter passed to method [cbe::delegate::ShareDelegate::onShareSuccess](#).

```
#include <PublishSuccess.h>
```

Public Member Functions

- **PublishSuccess** ([cbe::DefaultCtor](#))
- **PublishSuccess** ([cbe::Items](#) &&items)
- **operator bool** () const

Checks if current instance is valid.

Public Attributes

- [cbe::Items](#) items {}

12.104.1 Detailed Description

Convenience type that bundles the parameter passed to method [cbe::delegate::ShareDelegate::onShareSuccess](#).

12.104.2 Member Function Documentation

12.104.2.1 operator bool()

```
cbe::delegate::PublishSuccess::operator bool ( ) const [explicit]
```

Checks if current instance is valid.

Returns

`true` : is valid

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

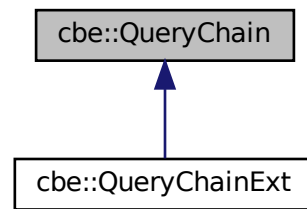
- include/cbe/delegate/PublishSuccess.h

12.105 cbe::QueryChain Class Reference

To do a search for [Object](#) combining more than one [Container](#) table.

```
#include <QueryChain.h>
```

Inheritance diagram for cbe::QueryChain:



Classes

- struct [Exception](#)

Public Types

- using **JoinDelegatePtr** = [delegate::JoinDelegatePtr](#)

Public Member Functions

- [QueryChain](#) **join** ([Container](#) containerToQuery, std::string key1, std::string key2, JoinDelegatePtr joinDelegate)
- [QueryChain](#) **join** ([Container](#) containerToQuery, std::string key1, std::string key2, [Filter](#) constraints, JoinDelegatePtr joinDelegate)
- [QueryChain](#) **join** ([Container](#) containerToQuery, std::string key1, std::string key2, [Container](#) containerForResults, JoinDelegatePtr joinDelegate)
- [QueryChain](#) **join** ([Container](#) containerToQuery, std::string key1, std::string key2, [Filter](#) constraints, [Container](#) containerForResults, JoinDelegatePtr joinDelegate)
- [QueryResult](#) **getQueryResult** () const
- **QueryChain** ([cbe::DefaultCtor](#))
- **operator bool** () const

Friends

- class **CloudBackend**
- class **Container**
- class **QueryChainSync**

12.105.1 Detailed Description

To do a search for [Object](#) combining more than one [Container](#) table.

[Async] [QueryChain](#) is used when a query() is carried out with a solely [QueryDelegate](#).

Anyway, if a subsequent [join\(\)](#) will be used, an additional [JoinDelegate](#) is required in the [join\(\)](#)-call.

12.105.2 Member Function Documentation

12.105.2.1 `getQueryResult()`

`QueryResult` `cbe::QueryChain::getQueryResult ( ) const`

Returns the `QueryResult` provided to the delegate after the the last `query()` or `join()` call. If this method is called before the delegate call, an empty `QueryResult` is returned.

12.105.2.2 `join()` [1/4]

```
QueryChain cbe::QueryChain::join (
    Container containerToQuery,
    std::string key1,
    std::string key2,
    Container containerForResults,
    JoinDelegatePtr joinDelegate )
```

Same as `join(Container, std::string, std::string, delegate::JoinDelegatePtr)`, but with an additional `Container` parameter `containerForResults`.

Parameters

<code>containerForResults</code>	Can be the <code>Container</code> from the previous query if desired items should be from that container.
----------------------------------	---

12.105.2.3 `join()` [2/4]

```
QueryChain cbe::QueryChain::join (
    Container containerToQuery,
    std::string key1,
    std::string key2,
    Filter constraints,
    Container containerForResults,
    JoinDelegatePtr joinDelegate )
```

Same as `join(Container, std::string, std::string, Container, delegate::JoinDelegatePtr)`, but with an additional `Filter` parameter `constraints`.

Parameters

<code>constraints</code>	A filter that provides any constraints for the query.
--------------------------	---

12.105.2.4 `join()` [3/4]

```
QueryChain cbe::QueryChain::join (
    Container containerToQuery,
    std::string key1,
    std::string key2,
    Filter constraints,
    JoinDelegatePtr joinDelegate )
```

Same as `join(Container, std::string, std::string, delegate::JoinDelegatePtr)`, but with an additional `Filter` parameter `constraints`.

Parameters

<code>constraints</code>	A filter that provides any constraints for the query.
--------------------------	---

12.105.2.5 join() [4/4]

```
QueryChain cbe::QueryChain::join (
    Container containerToQuery,
    std::string key1,
    std::string key2,
    JoinDelegatePtr joinDelegate )
```

Performs a join request in conjunction with a query, or previous join, to get related items from another container.

Asynchronous version of this service function.

Parameters

<i>containerToQuery</i>	The container of which the join shall be done with.
<i>key1</i>	The key from the previous query on which to perform the join.
<i>key2</i>	The key on objects in <i>containerToQuery</i> on which to perform the join
<i>joinDelegate</i>	Pointer to a JoinDelegate instance, implemented by the user, that receives the response of this query request. I.e., either of the JoinDelegate callback functions onJoinSuccess() or onJoinError() will be called.

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

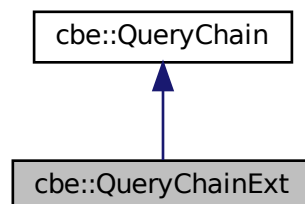
- include/cbe/QueryChain.h

12.106 cbe::QueryChainExt Class Reference

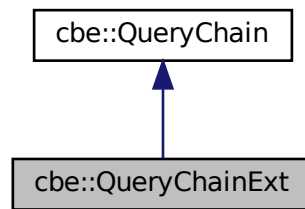
Extension of class [QueryChain](#).

```
#include <QueryChain.h>
```

Inheritance diagram for cbe::QueryChainExt:



Collaboration diagram for `cbe::QueryChainExt`:



Public Member Functions

- `QueryChainExt join` ([Container](#) containerToQuery, `std::string` key1, `std::string` key2)
- `QueryChainExt join` ([Container](#) containerToQuery, `std::string` key1, `std::string` key2, [Filter](#) constraints)
- `QueryChainExt join` ([Container](#) containerToQuery, `std::string` key1, `std::string` key2, [Container](#) containerForResults)
- `QueryChainExt join` ([Container](#) containerToQuery, `std::string` key1, `std::string` key2, [Filter](#) constraints, [Container](#) containerForResults)
- `QueryChain join` ([Container](#) containerToQuery, `std::string` key1, `std::string` key2, [JoinDelegatePtr](#) joinDelegate)
- `QueryChain join` ([Container](#) containerToQuery, `std::string` key1, `std::string` key2, [Filter](#) constraints, [JoinDelegatePtr](#) joinDelegate)
- `QueryChain join` ([Container](#) containerToQuery, `std::string` key1, `std::string` key2, [Container](#) containerForResults, [JoinDelegatePtr](#) joinDelegate)
- `QueryChain join` ([Container](#) containerToQuery, `std::string` key1, `std::string` key2, [Filter](#) constraints, [Container](#) containerForResults, [JoinDelegatePtr](#) joinDelegate)

Friends

- class `CloudBackend`
- class `Container`

Additional Inherited Members

12.106.1 Detailed Description

Extension of class [QueryChain](#).

[Async] `QueryChainExt` is used when the query is carried out with subsequent intended join in mind.

I.e., we have passed a [QueryJoinDelegate](#) to the previous call and that will then be reused in the trailing `join()`-call.

It is an extension of class [QueryChain](#) offering `join()` methods that do not require a new delegate.

Instead, these will use the delegate passed as argument in the previous invocation in the call chain — i.e.:

- `CloudBackend::query(ContainerId, delegate::QueryJoinDelegatePtr)` and overloads.
- `Container::query(delegate::QueryJoinDelegatePtr)` and overloads.
- `QueryChain::join(Container, std::string, std::string, JoinDelegatePtr)` and overloads.

12.106.2 Member Function Documentation

12.106.2.1 join() [1/8]

```
QueryChainExt cbe::QueryChainExt::join (
    Container containerToQuery,
    std::string key1,
    std::string key2 )
```

Performs a join request in line with function [join\(\)](#) in class [QueryChain](#).

If a call to this join function is chained with a previous call to a query, only such a query functions accepting a [QueryJoinDelegate](#) can be used as:

- [CloudBackend::query\(ContainerId,delegate::QueryJoinDelegatePtr\)](#)
- [CloudBackend::query\(ContainerId,Filter,delegate::QueryJoinDelegatePtr\)](#)
- [Container::query\(delegate::QueryJoinDelegatePtr\)](#)
- [Container::query\(Filter,delegate::QueryJoinDelegatePtr\)](#)

Asynchronous version of this service function.

See also

[QueryChain::join\(Container,std::string,std::string,delegate::JoinDelegatePtr\)](#)

12.106.2.2 join() [2/8]

```
QueryChainExt cbe::QueryChainExt::join (
    Container containerToQuery,
    std::string key1,
    std::string key2,
    Container containerForResults )
```

See also

[QueryChain::join\(Container,std::string,std::string,Container,delegate::JoinDelegatePtr\)](#)

12.106.2.3 join() [3/8]

```
QueryChain cbe::QueryChain::join
```

Same as [join\(Container,std::string,std::string,delegate::JoinDelegatePtr\)](#), but with an additional [Container](#) parameter `containerForResults`.

Parameters

<code>containerForResults</code>	Can be the Container from the previous query if desired items should be from that container.
----------------------------------	--

12.106.2.4 join() [4/8]

```
QueryChainExt cbe::QueryChainExt::join (
    Container containerToQuery,
    std::string key1,
    std::string key2,
    Filter constraints )
```

See also

[QueryChain::join\(Container,std::string,std::string,Filter,delegate::JoinDelegatePtr\)](#)

12.106.2.5 join() [5/8]

```
QueryChainExt cbe::QueryChainExt::join (
    Container containerToQuery,
    std::string key1,
    std::string key2,
    Filter constraints,
    Container containerForResults )
```

See also

[QueryChain::join\(Container,std::string,std::string,Filter,Container,delegate::JoinDelegatePtr\)](#)

12.106.2.6 join() [6/8]

```
QueryChain cbe::QueryChain::join
```

Same as [join\(Container,std::string,std::string,Container,delegate::JoinDelegatePtr\)](#), but with an additional [Filter](#) parameter `constraints`.

Parameters

<i>constraints</i>	A filter that provides any constraints for the query.
--------------------	---

12.106.2.7 join() [7/8]

```
QueryChain cbe::QueryChain::join
```

Same as [join\(Container,std::string,std::string,delegate::JoinDelegatePtr\)](#), but with an additional [Filter](#) parameter `constraints`.

Parameters

<i>constraints</i>	A filter that provides any constraints for the query.
--------------------	---

12.106.2.8 join() [8/8]

```
QueryChain cbe::QueryChain::join
```

Performs a join request in conjunction with a query, or previous join, to get related items from another container.
Asynchronous version of this service function.

Parameters

<i>containerToQuery</i>	The container of which the join shall be done with.
<i>key1</i>	The key from the previous query on which to perform the join.
<i>key2</i>	The key on objects in <code>containerToQuery</code> on which to perform the join
<i>joinDelegate</i>	Pointer to a JoinDelegate instance, implemented by the user, that receives the response of this query request. I.e., either of the JoinDelegate callback functions onJoinSuccess() or onJoinError() will be called.

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

- include/cbe/QueryChain.h

12.107 cbe::delegate::QueryDelegate Class Reference

```
#include <QueryDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::QueryResult](#)
- using **Error** = [QueryError](#)

Public Member Functions

- virtual void [onQuerySuccess](#) ([cbe::QueryResult](#) &&queryResult)=0
- virtual void [onQueryError](#) ([cbe::delegate::QueryError](#) &&error, [cbe::util::Context](#) &&context)=0

12.107.1 Detailed Description

Delegate class for the asynchronous version of methods:

- [CloudBackend::queryWithPath\(\)](#), and its overloads
- [CloudBackend::query\(\)](#), and its overloads
- [CloudBackend::search\(\)](#), and its overloads
- [Container::queryWithPath\(\)](#)
- [Container::query\(\)](#), and its overloads
- [Container::search\(\)](#), and its overloads

Example

```
#include "cbe/delegate/QueryDelegate.h"
~~~
#include <condition_variable>
#include <mutex>
~~~
class MyQueryDelegate : public cbe::delegate::QueryDelegate {
    std::mutex          mutex{};
    std::condition_variable conditionVariable{};
    // Indicates operation completed - success or failure
    bool                called = false;
public:
    cbe::QueryResult    queryResult{cbe::DefaultCtor{}};
    ErrorInfo           errorInfo{};
private:
    // Called upon successful query.
    void onQuerySuccess(cbe::QueryResult&& queryResult) final {
        {
            std::lock_guard<std::mutex> lock{mutex};
            // Change state of queryResult member to indicate success
            this->queryResult = std::move(queryResult);
            // Indicate operation completed, member queryResult indicates success
            called = true;
            // If delegate is reused, clear possibly error state
            errorInfo = ErrorInfo{};
        }
        conditionVariable.notify_one();
    } // onQuerySuccess()
    // Called upon a failed query() or join() call.
    void onQueryError(cbe::delegate::QueryError&& error,
                     cbe::util::Context&& context) final {
```

```

{
    std::lock_guard<std::mutex> lock{mutex};
    // Activate errorInfo member to indicate no-success
    errorInfo = ErrorInfo{std::move(context), std::move(error)};
    // Indicate operation completed, member object indicates failure
    called = true;
    // If delegate is reused, clear possibly success state
    queryResult = QueryResult{cbe::DefaultCtor{}};
}
conditionVariable.notify_one();
} // onQueryError()
public:
void waitForRsp() {
    std::unique_lock<std::mutex> lock{mutex};
    conditionVariable.wait(lock, [this] { return called; });
    // Reset called flag, so current delegate instance can be reused
    called = false;
} // waitForRsp()
}; // class MyQueryDelegate

```

Usage of the class above, see [Example of asynchronous query](#)

12.107.2 Member Function Documentation

12.107.2.1 onQueryError()

```

virtual void cbe::delegate::QueryDelegate::onQueryError (
    cbe::delegate::QueryError && error,
    cbe::util::Context && context ) [pure virtual]

```

Called upon a failed query() or join() call.

Parameters

<i>error</i>	Error information passed from CloudBackend SDK.
<i>context</i>	Additional context information about the original service call that has failed.

12.107.2.2 onQuerySuccess()

```

virtual void cbe::delegate::QueryDelegate::onQuerySuccess (
    cbe::QueryResult && queryResult ) [pure virtual]

```

Called upon successful query.

Parameters

<i>queryResult</i>	Instance of a QueryResult containing the result set.
--------------------	--

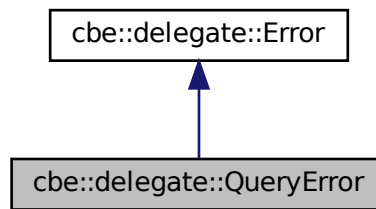
The documentation for this class was generated from the following file:

- include/cbe/delegate/QueryDelegate.h

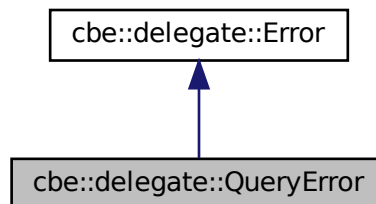
12.108 cbe::delegate::QueryError Class Reference

```
#include <QueryError.h>
```

Inheritance diagram for cbe::delegate::QueryError:



Collaboration diagram for cbe::delegate::QueryError:



Public Member Functions

- **QueryError** ([ErrorCode](#) `errorCode`, std::string &&`reason`, std::string &&`message`)
- **QueryError** ([Filter](#) &&`filter`, [ErrorCode](#) `errorCode`, std::string &&`reason`, std::string &&`message`)
- bool **hasFilter** () const
- const [Filter](#) & **getFilter** () const

Friends

- std::ostream & **operator**<< (std::ostream &os, const [QueryError](#) &error)

Additional Inherited Members

12.108.1 Detailed Description

Contains error information delivered from CloudBackend SDK i connection with a failed query, or join
The documentation for this class was generated from the following file:

- include/cbe/delegate/QueryError.h

12.109 cbe::delegate::QueryJoinDelegate Class Reference

```
#include <QueryJoinDelegate.h>
```

Public Types

- using **Success** = [cbe::QueryResult](#)
- using **QueryJoinError** = [QueryError](#)
- using **Error** = [QueryJoinError](#)
- using **ErrorInfo** = [QueryDelegate::ErrorInfo](#)

Public Member Functions

- virtual void [onQueryJoinSuccess](#) ([cbe::QueryResult](#) &&queryResult)=0
- virtual void [onQueryJoinError](#) ([QueryJoinError](#) &&error, [cbe::util::Context](#) &&context)=0

12.109.1 Detailed Description

Delegate callback interface for asynchronous methods:

- [CloudBackend::query\(ContainerId,delegate::QueryJoinDelegatePtr\)](#) and overloads.
- [Container::query\(delegate::QueryJoinDelegatePtr\)](#) and overloads.

12.109.2 Member Typedef Documentation

12.109.2.1 ErrorInfo

using [cbe::delegate::QueryJoinDelegate::ErrorInfo](#) = [QueryDelegate::ErrorInfo](#)

Contains all information about a failed query.

12.109.3 Member Function Documentation

12.109.3.1 onQueryJoinError()

```
virtual void cbe::delegate::QueryJoinDelegate::onQueryJoinError (
    QueryJoinError && error,
    cbe::util::Context && context ) [pure virtual]
```

Called upon a failed query() or join() call.

Parameters

<i>error</i>	Error information passed from CloudBackend SDK.
<i>context</i>	Additional context information about the original service call that has failed.

12.109.3.2 onQueryJoinSuccess()

```
virtual void cbe::delegate::QueryJoinDelegate::onQueryJoinSuccess (
    cbe::QueryResult && queryResult ) [pure virtual]
```

Called upon successful query.

Parameters

<i>queryResult</i>	Instance of a QueryResult containing the result set.
--------------------	--

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

- include/cbe/delegate/QueryJoinDelegate.h

12.110 cbe::delegate::QueryLoaded Struct Reference

Public Member Functions

- virtual void [onQuerySuccess](#) ([cbe::QueryResult](#) &&queryResult)=0

12.110.1 Member Function Documentation

12.110.1.1 onQuerySuccess()

```
virtual void cbe::delegate::QueryLoaded::onQuerySuccess (
    cbe::QueryResult && queryResult ) [pure virtual]
```

Called upon successful query.

Parameters

<i>queryResult</i>	Instance of a QueryResult containing the result set.
--------------------	--

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

- include/cbe/delegate/QueryLoaded.h

12.111 cbe::QueryResult Class Reference

resultset of data retrieved.

```
#include <QueryResult.h>
```

Public Types

- using [ItemsSnapshot](#) = std::vector< [Item](#) >
A vector of items to hold a snapshot.

Public Member Functions

- [Filter filter](#) () const
- [ItemsSnapshot getItemsSnapshot](#) () const
Returns a copy of a vector containing the items for this [QueryResult](#).
- std::uint64_t [itemsLoaded](#) () const
- std::uint64_t [totalCount](#) () const
- std::uint64_t [objectsLoaded](#) () const
- std::uint64_t [containersLoaded](#) () const
- bool [containsItem](#) ([ItemId](#) itemId) const
- [QueryResult](#) ([cbe::DefaultCtor](#))
Default constructor.
- [operator bool](#) () const
Checks if the current instance is real.

Friends

- class [CloudBackend](#)
- class [Container](#)

- class **QueryChain**
- class **QueryChainSync**
- class **ShareManager**
- class **SubscribeManager**
- `std::ostream & operator<< (std::ostream &os, const QueryResult &queryResult)`
- `CBI::ItemDelegatePtr delegate::createCbiQueryDelegate (delegate::QueryDelegatePtr, cbe::util::Context &&)`
- `CBI::ItemDelegatePtr delegate::createCbiQueryDelegate (delegate::QueryJoinDelegatePtr, cbe::util::Context &&)`
- `CBI::ItemDelegatePtr delegate::createCbiJoinDelegate (delegate::JoinDelegatePtr, cbe::util::Context &&)`

12.111.1 Detailed Description

resultset of data retrieved.

12.111.2 Constructor & Destructor Documentation

12.111.2.1 [QueryResult\(\)](#)

```
cbe::QueryResult::QueryResult (
    cbe::DefaultCtor )
```

Default constructor.

Construct a new object with the `DefaultCtor` to enable the [operator bool\(\)](#) test

12.111.3 Member Function Documentation

12.111.3.1 [containersLoaded\(\)](#)

```
std::uint64_t cbe::QueryResult::containersLoaded ( ) const
```

Returns number of `containers` loaded in to the query result.

12.111.3.2 [containsItem\(\)](#)

```
bool cbe::QueryResult::containsItem (
    ItemId itemId ) const
```

Checks if the [Item](#) with `id` is in the result-set.

Parameters

<i>itemId</i>	id number of the item asked for
---------------	---------------------------------

12.111.3.3 [filter\(\)](#)

```
Filter cbe::QueryResult::filter ( ) const
```

Returns a copy of the filter used for query.

12.111.3.4 [getItemsSnapshot\(\)](#)

```
ItemsSnapshot cbe::QueryResult::getItemsSnapshot ( ) const
```

Returns a copy of a vector containing the items for this [QueryResult](#).

The [QueryResult](#) will be updated when new data comes in, but the returned copy will not.

Note

If iterating, make sure to create a variable for a local copy.

Returns

The items matching the query.

12.111.3.5 itemsLoaded()

```
std::uint64_t cbe::QueryResult::itemsLoaded ( ) const
```

Number of items loaded in current [QueryResult](#).

12.111.3.6 objectsLoaded()

```
std::uint64_t cbe::QueryResult::objectsLoaded ( ) const
```

Returns number of objects loaded in to the query result.

12.111.3.7 operator bool()

```
cbe::QueryResult::operator bool ( ) const [explicit]
```

Checks if the current instance is real.
An "unreal" instance implies typically a failed event.
Relies on the Default constructor [QueryResult\(cbe::DefaultCtor\)](#)

Returns

`true` : is real
`false` : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

12.111.3.8 totalCount()

```
std::uint64_t cbe::QueryResult::totalCount ( ) const
```

total number of items in the cloud matching the query result. This may be more than loaded.
The documentation for this class was generated from the following file:

- include/cbe/QueryResult.h

12.112 cbe::delegate::container::RemoveDelegate Class Reference

```
#include <RemoveDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [RemoveSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onRemoveSuccess](#) ([cbe::ItemId](#) containerId, std::string name)=0
- virtual void [onRemoveError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.112.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Container::remove](#)

12.112.2 Member Function Documentation

12.112.2.1 onRemoveError()

```
virtual void cbe::delegate::container::RemoveDelegate::onRemoveError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.112.2.2 onRemoveSuccess()

```
virtual void cbe::delegate::container::RemoveDelegate::onRemoveSuccess (
    cbe::ItemId containerId,
    std::string name ) [pure virtual]
```

Called upon successful Remove.

Parameters

<i>containerId</i>	Id of the cbe::Container being removed.
<i>name</i>	Name of container that is being removed.

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

- [include/cbe/delegate/container/RemoveDelegate.h](#)

12.113 cbe::delegate::group::RemoveDelegate Class Reference

```
#include <RemoveDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [RemoveSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onRemoveSuccess](#) ([cbe::GroupId](#) groupId, std::string &&groupName)=0
- virtual void [onRemoveError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.113.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Group::remove](#)

12.113.2 Member Function Documentation

12.113.2.1 onRemoveError()

```
virtual void cbe::delegate::group::RemoveDelegate::onRemoveError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.113.2.2 onRemoveSuccess()

```
virtual void cbe::delegate::group::RemoveDelegate::onRemoveSuccess (
    cbe::GroupId groupId,
    std::string && groupName ) [pure virtual]
```

Called upon successful remove.

Parameters

<i>groupId</i>	Id of the cbe::Group being removed.
<i>groupName</i>	Name of the group being removed.

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

- include/cbe/delegate/group/RemoveDelegate.h

12.114 cbe::delegate::object::RemoveDelegate Class Reference

```
#include <RemoveDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [RemoveSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onRemoveSuccess](#) ([cbe::ItemId](#) objectId, std::string name)=0
- virtual void [onRemoveError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.114.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Object::remove](#)

12.114.2 Member Function Documentation

12.114.2.1 onRemoveError()

```
virtual void cbe::delegate::object::RemoveDelegate::onRemoveError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.114.2.2 onRemoveSuccess()

```
virtual void cbe::delegate::object::RemoveDelegate::onRemoveSuccess (
    cbe::ItemId objectId,
    std::string name ) [pure virtual]
```

Called upon successful object remove.

Parameters

<i>objectId</i>	Id of object being removed.
<i>name</i>	Name of object being removed.

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

- include/cbe/delegate/object/RemoveDelegate.h

12.115 cbe::delegate::RemoveRoleDelegate Class Reference

```
#include <RemoveRoleDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::RoleId](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onRemoveRoleSuccess](#) ([cbe::RoleId](#) &&roleId)=0
- virtual void [onRemoveRoleError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.115.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Role::removeRole](#)

12.115.2 Member Function Documentation

12.115.2.1 onRemoveRoleError()

```
virtual void cbe::delegate::RemoveRoleDelegate::onRemoveRoleError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.115.2.2 onRemoveRoleSuccess()

```
virtual void cbe::delegate::RemoveRoleDelegate::onRemoveRoleSuccess (
    cbe::RoleId && roleId ) [pure virtual]
```

Called upon successful removal of the [Role](#).

Parameters

Role	the role id of the removed role.
----------------------	----------------------------------

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

- include/cbe/delegate/RemoveRoleDelegate.h

12.116 cbe::delegate::RemoveRoleMemberDelegate Class Reference

```
#include <RemoveRoleMemberDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [MemberId](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onRemoveRoleMemberSuccess](#) ([MemberId](#) memberId)=0
- virtual void [onRemoveRoleMemberError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.116.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Group::RemoveMember](#)

12.116.2 Member Function Documentation

12.116.2.1 onRemoveRoleMemberError()

```
virtual void cbe::delegate::RemoveRoleMemberDelegate::onRemoveRoleMemberError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.116.2.2 onRemoveRoleMemberSuccess()

```
virtual void cbe::delegate::RemoveRoleMemberDelegate::onRemoveRoleMemberSuccess (
    MemberId memberId ) [pure virtual]
```

Called upon successful RemoveMember.

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

- include/cbe/delegate/RemoveRoleMemberDelegate.h

12.117 cbe::delegate::container::RemoveSuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::container::RemoveDelegate::onRemoveSuccess](#).
`#include <RemoveDelegate.h>`

Public Member Functions

- **RemoveSuccess** ([cbe::DefaultCtor](#))
- **RemoveSuccess** ([cbe::ItemId](#) containerId, std::string name)

Public Attributes

- [cbe::ItemId](#) **containerId** {}
- std::string **name** {}

12.117.1 Detailed Description

Convenience type that bundles all parameters passed to method [cbe::delegate::container::RemoveDelegate::onRemoveSuccess](#).
The documentation for this class was generated from the following file:

- include/cbe/delegate/container/RemoveDelegate.h

12.118 cbe::delegate::group::RemoveSuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::group::RemoveDelegate::onRemoveSuccess](#).
`#include <RemoveDelegate.h>`

Public Member Functions

- **RemoveSuccess** ([cbe::DefaultCtor](#))
- **RemoveSuccess** ([cbe::GroupId](#) groupId, std::string &&groupName)

Public Attributes

- [cbe::GroupId](#) **groupId** {}
- std::string **groupName** {}

12.118.1 Detailed Description

Convenience type that bundles all parameters passed to method [cbe::delegate::group::RemoveDelegate::onRemoveSuccess](#).
The documentation for this class was generated from the following file:

- include/cbe/delegate/group/RemoveDelegate.h

12.119 cbe::delegate::object::RemoveSuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::object::onRemoveSuccess](#).
`#include <RemoveDelegate.h>`

Public Member Functions

- **RemoveSuccess** ([cbe::DefaultCtor](#))
- **RemoveSuccess** ([cbe::ItemId](#) objectId, std::string name)
- [operator bool](#) () const

Checks if current instance is valid.

Public Attributes

- [cbe::ItemId](#) **objectId** {}
- `std::string` **name** {}

12.119.1 Detailed Description

Convenience type that bundles all parameters passed to method `cbe::delegate::object::onRemoveSuccess`.

12.119.2 Member Function Documentation

12.119.2.1 operator bool()

```
cbe::delegate::object::RemoveSuccess::operator bool ( ) const [explicit]
```

Checks if current instance is valid.

Returns

`true`: is valid

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

- `include/cbe/delegate/object/RemoveDelegate.h`

12.120 cbe::delegate::container::RenameDelegate Class Reference

```
#include <RenameDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Container](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onRenameSuccess](#) ([cbe::Container](#) &&container)=0
- virtual void [onRenameError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.120.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Container::rename](#)

12.120.2 Member Function Documentation

12.120.2.1 onRenameError()

```
virtual void cbe::delegate::container::RenameDelegate::onRenameError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.120.2.2 onRenameSuccess()

```
virtual void cbe::delegate::container::RenameDelegate::onRenameSuccess (
    cbe::Container && container ) [pure virtual]
```

Called upon successful Rename.

Parameters

<i>container</i>	Instance of container that is being Renamed.
------------------	--

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

- include/cbe/delegate/container/RenameDelegate.h

12.121 cbe::delegate::group::RenameDelegate Class Reference

```
#include <RenameDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [RenameSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onRenameSuccess](#) ([cbe::Group](#) &&group, std::string &&newName)=0
- virtual void [onRenameError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.121.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Group::rename](#)

12.121.2 Member Function Documentation

12.121.2.1 onRenameError()

```
virtual void cbe::delegate::group::RenameDelegate::onRenameError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.121.2.2 onRenameSuccess()

```
virtual void cbe::delegate::group::RenameDelegate::onRenameSuccess (
    cbe::Group && group,
    std::string && newName ) [pure virtual]
```

Called upon successful rename.

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

- include/cbe/delegate/group/RenameDelegate.h

12.122 cbe::delegate::object::RenameDelegate Class Reference

```
#include <RenameDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Object](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onRenameSuccess](#) ([cbe::Object](#) &&object)=0
- virtual void [onRenameError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.122.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Object::rename](#)

12.122.2 Member Function Documentation

12.122.2.1 onRenameError()

```
virtual void cbe::delegate::object::RenameDelegate::onRenameError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.122.2.2 onRenameSuccess()

```
virtual void cbe::delegate::object::RenameDelegate::onRenameSuccess (
    cbe::Object && object ) [pure virtual]
```

Called upon successful Rename.

Parameters

<i>object</i>	Instance of object that are being renamed.
---------------	--

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

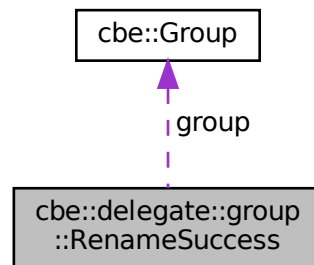
- include/cbe/delegate/object/RenameDelegate.h

12.123 cbe::delegate::group::RenameSuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::group::RenameDelegate::onRenameSuccess](#).

```
#include <RenameSuccess.h>
```

Collaboration diagram for `cbe::delegate::group::RenameSuccess`:



Public Member Functions

- **RenameSuccess** ([cbe::DefaultCtor](#))
- **RenameSuccess** ([cbe::Group](#) &&group, std::string &&newName)
- **operator bool** () const
Checks if current instance is valid.

Public Attributes

- [cbe::Group](#) **group** {[cbe::DefaultCtor](#){}}
- std::string **newName** {}

12.123.1 Detailed Description

Convenience type that bundles all parameters passed to method [cbe::delegate::group::RenameDelegate::onRenameSuccess](#).

12.123.2 Member Function Documentation

12.123.2.1 operator bool()

```
cbe::delegate::group::RenameSuccess::operator bool ( ) const [explicit]
```

Checks if current instance is valid.

Returns

`true`: is valid

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

- include/cbe/delegate/group/RenameSuccess.h

12.124 cbe::Request Struct Reference

```
#include <Member.h>
```

Public Member Functions

- **Request** ([cbe::UserId](#) userId, std::string alias, std::string applicationComment, [cbe::Date](#) date)

Public Attributes

- [cbe::UserId](#) `userId` {}
- `std::string` `alias` {}
- `std::string` `applicationComment` {}
- [cbe::Date](#) `date` {}

12.124.1 Detailed Description

Contains information on a request for a [Member](#) to join a group.

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

- `include/cbe/Member.h`

12.125 cbe::Role Class Reference

User role information.

```
#include <Role.h>
```

Public Types

- using [ListMembersDelegatePtr](#) = [delegate::ListMembersDelegatePtr](#)
- using [AddRoleMemberDelegatePtr](#) = [delegate::AddRoleMemberDelegatePtr](#)
- using [RemoveRoleMemberDelegatePtr](#) = [delegate::RemoveRoleMemberDelegatePtr](#)

Public Member Functions

- `std::string` [name](#) () const
- [cbe::RoleId](#) [id](#) () const
- [cbe::GroupId](#) [groupId](#) () const
- void [listMembers](#) ([ListMembersDelegatePtr](#) delegate)
Lists the members of this role,.
- void [addRoleMember](#) ([MemberId](#) memberId, [AddRoleMemberDelegatePtr](#) delegate)
Adds a member to this role.
- void [removeRoleMember](#) ([MemberId](#) memberId, [RemoveRoleMemberDelegatePtr](#) delegate)
Removes a member to this role.
- [Role](#) ([cbe::DefaultCtor](#))
Default constructor.
- [operator bool](#) () const
Checks if the current instance is real.

Friends

- class [Group](#)

12.125.1 Detailed Description

User role information.

12.125.2 Member Typedef Documentation

12.125.2.1 AddRoleMemberDelegatePtr

using `cbe::Role::AddRoleMemberDelegatePtr = delegate::AddRoleMemberDelegatePtr`

Pointer to `cbe::delegate::ListMembersDelegate` that is passed into asynchronous version of method `listMembers()`

Pointer to `AddRoleMemberDelegate` that is passed into:

- `cbe::Group::AddRoleMember(AddRoleMemberDelegatePtr).`

12.125.2.2 ListMembersDelegatePtr

using `cbe::Role::ListMembersDelegatePtr = delegate::ListMembersDelegatePtr`

Pointer to `cbe::delegate::ListMembersDelegate` that is passed into asynchronous version of method `listMembers()`

12.125.2.3 RemoveRoleMemberDelegatePtr

using `cbe::Role::RemoveRoleMemberDelegatePtr = delegate::RemoveRoleMemberDelegatePtr`

Pointer to `cbe::delegate::AddRoleMemberDelegate` that is passed into asynchronous version of method `AddRoleMember()` Pointer to `RemoveRoleMember` that is passed into:

`Group::RemoveRoleMember(std::string, RemoveRoleMemberDelegatePtr).`

12.125.3 Constructor & Destructor Documentation

12.125.3.1 Role()

```
cbe::Role::Role (
    cbe::DefaultCtor )
```

Default constructor.

Pointer to `cbe::delegate::RemoveRoleMember` that is passed into asynchronous version of method `RemoveRoleMember()`

Construct a new object with the `DefaultCtor` to enable the `operator bool()` test

12.125.4 Member Function Documentation

12.125.4.1 addRoleMember()

```
void cbe::Role::addRoleMember (
    MemberId memberId,
    AddRoleMemberDelegatePtr delegate )
```

Adds a member to this role.

Parameters

<i>memberId</i>	The id of the member to add to the role.
<i>delegate</i>	Pointer to a <code>delegate::AddRoleMemberDelegate</code> instance that is implemented by the user.

12.125.4.2 groupId()

```
cbe::GroupId cbe::Role::groupId ( ) const
```

Id of the group the role belongs to.

12.125.4.3 id()

```
cbe::RoleId cbe::Role::id ( ) const
```

Id of the role.

12.125.4.4 listMembers()

```
void cbe::Role::listMembers (
    ListMembersDelegatePtr delegate )
```

Lists the members of this role,.

Parameters

<i>delegate</i>	Pointer to a delegate::ListMembersDelegate instance that is implemented by the user.
-----------------	--

12.125.4.5 name()

```
std::string cbe::Role::name ( ) const
```

Name of the role.

12.125.4.6 operator bool()

```
cbe::Role::operator bool ( ) const [explicit]
```

Checks if the current instance is real.
An "unreal" instance implies typically a failed event.
Relies on the Default constructor [Role\(cbe::DefaultCtor\)](#)

Returns

`true` : is real
`false` : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

12.125.4.7 removeRoleMember()

```
void cbe::Role::removeRoleMember (
    MemberId memberId,
    RemoveRoleMemberDelegatePtr delegate )
```

Removes a member to this role.

Parameters

<i>memberId</i>	The id of the member to remove from the role.
<i>delegate</i>	Pointer to a delegate::RemoveRoleMember instance that is implemented by the user.

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

- `include/cbe/Role.h`

12.126 cbe::delegate::SearchGroupsDelegate Class Reference

```
#include <SearchGroupsDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::GroupQueryResult](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onSearchGroupsSuccess](#) ([cbe::GroupQueryResult](#) &&queryResult)=0
- virtual void [onSearchGroupsError](#) ([delegate::Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.126.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::GroupManager::searchGroups\(\)](#)

12.126.2 Member Function Documentation

12.126.2.1 onSearchGroupsError()

```
virtual void cbe::delegate::SearchGroupsDelegate::onSearchGroupsError (
    delegate::Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.126.2.2 onSearchGroupsSuccess()

```
virtual void cbe::delegate::SearchGroupsDelegate::onSearchGroupsSuccess (
    cbe::GroupQueryResult && queryResult ) [pure virtual]
```

Called upon successful search.

Parameters

<i>queryResult</i>	Ref. instance of cbe::GroupQueryResult holding the result of the search.
--------------------	--

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

- include/cbe/delegate/SearchGroupsDelegate.h

12.127 cbe::ShareData Struct Reference

```
#include <Utility.h>
```

Public Member Functions

- template<class ShareDataT >
ShareData (ShareDataT &&rh)
- **ShareData** (uint64_t id, bool isUserId)

Public Attributes

- uint64_t **id**
- bool **isUserId**

12.127.1 Detailed Description

This is the data used from shares(shareId) to keep track of user/group-id relating to a shareId

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

- include/cbe/Utility.h

12.128 cbe::delegate::ShareDelegate Class Reference

```
#include <ShareDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [ShareSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onShareSuccess](#) ([cbe::ShareId](#) shareId)=0
- virtual void [onShareError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.128.1 Detailed Description

Delegate class for the asynchronous version of methods:

- [cbe::Container::share](#)
- [cbe::Object::share](#)

12.128.2 Member Function Documentation

12.128.2.1 onShareError()

```
virtual void cbe::delegate::ShareDelegate::onShareError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.128.2.2 onShareSuccess()

```
virtual void cbe::delegate::ShareDelegate::onShareSuccess (
    cbe::ShareId shareId ) [pure virtual]
```

Called upon successful share.

Parameters

<i>shareId</i>	Id of the share. Can be used when unsharing. See Asynchronous version of cbe::Object::unshare()
----------------	---

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

- include/cbe/delegate/ShareDelegate.h

12.129 cbe::ShareManager Class Reference

Managing Shares.

```
#include <ShareManager.h>
```

Public Types

- using [ListSharesDelegatePtr](#) = [delegate::ListSharesDelegatePtr](#)

Public Member Functions

- void [listAvailableShares](#) ([ListSharesDelegatePtr](#) delegate)
Lists the shares exposed by other users to current user. This will give specific share information.
- void [listMyShares](#) ([ListSharesDelegatePtr](#) delegate)
Lists the shares exposed by current user. This will give specific share information.
- [ShareManager](#) ([cbe::DefaultCtor](#))
Default constructor.
- [operator bool](#) () const
Checks if the current instance is real.

Friends

- class [CloudBackend](#)

12.129.1 Detailed Description

Managing Shares.

This class represents a list of Shares.

12.129.2 Member Typedef Documentation

12.129.2.1 ListSharesDelegatePtr

```
using cbe::ShareManager::ListSharesDelegatePtr = delegate::ListSharesDelegatePtr
```

Pointer to ListSharesDelegate that is passed into:

- [ShareManager::listAvailableShares\(ListSharesDelegatePtr\)](#)
- [ShareManager::listMyShares\(ListSharesDelegatePtr\)](#)

12.129.3 Constructor & Destructor Documentation

12.129.3.1 ShareManager()

```
cbe::ShareManager::ShareManager (  
    cbe::DefaultCtor )
```

Default constructor.

Pointer to [cbe::delegate::ListSharesDelegate](#) that is passed into asynchronous version of method [listMyShares\(\)](#)

Construct a new object with the `DefaultCtor` to enable the [operator bool\(\)](#) test

12.129.4 Member Function Documentation

12.129.4.1 listAvailableShares()

```
void cbe::ShareManager::listAvailableShares (
    ListSharesDelegatePtr delegate )
```

Lists the shares exposed by other users to current user. This will give specific share information.

Parameters

<i>delegate</i>	Pointer to a delegate::ListSharesDelegate instance that is implemented by the user.
-----------------	---

12.129.4.2 listMyShares()

```
void cbe::ShareManager::listMyShares (
    ListSharesDelegatePtr delegate )
```

Lists the shares exposed by current user. This will give specific share information.

Pointer to [cbe::delegate::ListSharesDelegate](#) that is passed into asynchronous version of method [listAvailableShares\(\)](#)

Parameters

<i>delegate</i>	Pointer to a delegate::ListSharesDelegate instance that is implemented by the user.
-----------------	---

12.129.4.3 operator bool()

```
cbe::ShareManager::operator bool ( ) const [explicit]
```

Checks if the current instance is real.

An "unreal" instance implies typically a failed event.

Relies on the Default constructor [ShareManager\(cbe::DefaultCtor\)](#)

Returns

`true` : is real

`false` : unreal; got nullptr; if current instance is unbound/undefined. I.e., if it is only default constructed.

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

- include/cbe/ShareManager.h

12.130 cbe::delegate::ShareSuccess Class Reference

Convenience type that bundles the parameter passed to method [cbe::delegate::ShareDelegate::onShareSuccess](#).

```
#include <ShareDelegate.h>
```

Public Member Functions

- [ShareSuccess](#) ([cbe::DefaultCtor](#))
- [ShareSuccess](#) ([cbe::ShareId](#) shareId)
- [operator bool](#) () const

Checks if current instance is valid.

Public Attributes

- [cbe::ShareId](#) shareId = 0

12.130.1 Detailed Description

Convenience type that bundles the parameter passed to method [cbe::delegate::ShareDelegate::onShareSuccess](#).

12.130.2 Member Function Documentation

12.130.2.1 operator bool()

```
cbe::delegate::ShareSuccess::operator bool ( ) const [explicit]
```

Checks if current instance is valid.

Returns

true: is valid

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

- include/cbe/delegate/ShareDelegate.h

12.131 cbe::Stream Class Reference

A data file attached to [Object](#).

```
#include <Stream.h>
```

Public Attributes

- [cbe::StreamId](#) streamId {}
- std::size_t length {}

Friends

- class [Object](#)

12.131.1 Detailed Description

A data file attached to [Object](#).

An object kan have zero, one or many streams.

12.131.2 Member Data Documentation

12.131.2.1 length

```
std::size_t cbe::Stream::length {}
```

the size in Bytes

12.131.2.2 streamId

```
cbe::StreamId cbe::Stream::streamId {}
```

the id number of the stream

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

- include/cbe/Stream.h

12.132 cbe::Subscribe Class Reference

Managing a subscribed [Item](#).

```
#include <Subscribe.h>
```

Public Member Functions

- [cbe::Date](#) [getDate](#) () const
- std::string [getTitle](#) () const
- std::string [getDescription](#) () const
- std::string [getPassword](#) () const
- [cbe::PublishAccess](#) [getSecurity](#) () const
- [cbe::PublishVisibility](#) [getPrivacy](#) () const
- [cbe::Subscribeld](#) [getSubscribeld](#) () const
- [cbe::PublishId](#) [getPublishId](#) () const
- [cbe::UserId](#) [getOwner](#) () const
- void [unSubscribe](#) ()
- **Subscribe** ([cbe::DefaultCtor](#))
- **operator bool** () const

Friends

- class **CloudBackend**
- class **Item**

12.132.1 Detailed Description

Managing a subscribed [Item](#).

To inspect the settings of a subscription.

12.132.2 Member Function Documentation

12.132.2.1 getDate()

```
cbe::Date cbe::Subscribe::getDate ( ) const
```

Gets the date as unix epoch number

12.132.2.2 getDescription()

```
std::string cbe::Subscribe::getDescription ( ) const
```

Gets the description

12.132.2.3 getOwner()

```
cbe::UserId cbe::Subscribe::getOwner ( ) const
```

Gets the owner id number

12.132.2.4 getPassword()

```
std::string cbe::Subscribe::getPassword ( ) const
```

Checks if a password has been set.

Returns

"true" : if a password has been set

"" : empty string if not.

12.132.2.5 getPrivacy()

```
cbe::PublishVisibility cbe::Subscribe::getPrivacy ( ) const
```

Gets the privacy see [cbe::PublishVisibility](#) enum

12.132.2.6 getPublishId()

`cbe::PublishId cbe::Subscribe::getPublishId ( ) const`
 Gets the publish id number

12.132.2.7 getSecurity()

`cbe::PublishAccess cbe::Subscribe::getSecurity ( ) const`
 Gets the security see [cbe::PublishAccess](#) enum

12.132.2.8 getSubscribeId()

`cbe::SubscribeId cbe::Subscribe::getSubscribeId ( ) const`
 Gets the subscribe id number

12.132.2.9 getTitle()

`std::string cbe::Subscribe::getTitle ( ) const`
 Gets the title

12.132.2.10 unSubscribe()

`void cbe::Subscribe::unSubscribe ( )`
 unSubscribe to this subscription.
 The documentation for this class was generated from the following file:

- `include/cbe/Subscribe.h`

12.133 cbe::delegate::SubscribeDelegate Class Reference

`#include <SubscribeDelegate.h>`

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Items](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onSubscribeSuccess](#) ([cbe::Items](#) &&success)=0
- virtual void [onSubscribeError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.133.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::SubscribeManager::subscribe](#)

12.133.2 Member Function Documentation

12.133.2.1 onSubscribeError()

```
virtual void cbe::delegate::SubscribeDelegate::onSubscribeError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.133.2.2 onSubscribeSuccess()

```
virtual void cbe::delegate::SubscribeDelegate::onSubscribeSuccess (
    cbe::Items && success ) [pure virtual]
```

Called upon successful subscribe.

Parameters

<i>success</i>	Instance of Items that are being subscribed.
----------------	--

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

- include/cbe/delegate/SubscribeDelegate.h

12.134 cbe::SubscribeManager Class Reference

For managing subscriptions.

```
#include <SubscribeManager.h>
```

Public Types

- using [GetSubscriptionsDelegatePtr](#) = [delegate::GetSubscriptionsDelegatePtr](#)
- using [SubscribeDelegatePtr](#) = [delegate::SubscribeDelegatePtr](#)

Public Member Functions

- void [getSubscriptions](#) ([GetSubscriptionsDelegatePtr](#) subscribeDelegate)
List your current subscriptions.
- void [subscribe](#) ([cbe::UserId](#) sharingUserId, std::string sharingUserName, [cbe::PublishId](#) publishId, std::string publishName, std::string password, std::string subscribeName, [SubscribeDelegatePtr](#) subscribeDelegate)
Subscribes to a [Publish](#) using all parameters.
- void [subscribe](#) (std::string sharingUserName, std::string publishName, [SubscribeDelegatePtr](#) subscribeDelegate)
Subscribes to a [Publish](#) using names.
- void [subscribe](#) ([cbe::UserId](#) sharingUserId, [cbe::PublishId](#) publishId, std::string subscribeName, [SubscribeDelegatePtr](#) subscribeDelegate)
Subscribes to a [Publish](#) using ids.
- **SubscribeManager** ([cbe::DefaultCtor](#))
- **operator bool** () const

Friends

- class **CloudBackend**

12.134.1 Detailed Description

For managing subscriptions.

12.134.2 Member Typedef Documentation

12.134.2.1 GetSubscriptionsDelegatePtr

using `cbe::SubscribeManager::GetSubscriptionsDelegatePtr = delegate::GetSubscriptionsDelegatePtr`
 Pointer to `GetSubscriptionsDelegate` that is passed into asynchronous version of method:

- `getSubscriptions()`

12.134.2.2 SubscribeDelegatePtr

using `cbe::SubscribeManager::SubscribeDelegatePtr = delegate::SubscribeDelegatePtr`
 Pointer to `cbe::delegate::GetSubscriptionsDelegate` that is passed into asynchronous version of method
`getSubscriptions()` Pointer to `SubscribeDelegate` that is passed into asynchronous version of method:

- `subscribe()`

12.134.3 Member Function Documentation

12.134.3.1 getSubscriptions()

```
void cbe::SubscribeManager::getSubscriptions (
    GetSubscriptionsDelegatePtr subscribeDelegate )
```

List your current subscriptions.

Listing is done independently of where in the actual directory tree the files are located.

Parameters

<i>subscribeDelegate</i>	Pointer to a <code>delegate::GetSubscriptionsDelegate</code> instance that is implemented by the user.
--------------------------	--

12.134.3.2 subscribe() [1/3]

```
void cbe::SubscribeManager::subscribe (
    cbe::UserId sharingUserId,
    cbe::PublishId publishId,
    std::string subscribeName,
    SubscribeDelegatePtr subscribeDelegate ) [inline]
```

Subscribes to a `Publish` using ids.

Overload of `subscribe()`

Parameters

<i>sharingUserId</i>	User id of the owner of the publish
<i>publishId</i>	publish id (publish id)
<i>subscribeName</i>	Name of created subscribe (usually same as the publish)
<i>subscribeDelegate</i>	Gets notified of the result

12.134.3.3 subscribe() [2/3]

```
void cbe::SubscribeManager::subscribe (
    cbe::UserId sharingUserId,
    std::string sharingUserName,
    cbe::PublishId publishId,
```

```
std::string publishName,
std::string password,
std::string subscribeName,
SubscribeDelegatePtr subscribeDelegate )
```

Subscribes to a [Publish](#) using all parameters.

Subscribe to a [Publish](#) that was issued by some other user (or yourself).

A [Publish](#) must not require a password, if required it should be provided by the publisher.

The retrieved subscription may be revoked with [cbe::Object::unsubscribe\(\)](#) or [cbe::Container::unsubscribe\(\)](#).

Subscribes to a publish shared by some other user (or yourself).

Parameters

<i>sharingUserId</i>	User id of the owner of the publish or 0 if sharingUserName should be used instead
<i>sharingUserName</i>	User name of the owner of the publish or empty if sharingUserId should be used instead
<i>publishId</i>	Publish id or 0 if publishName should be used instead
<i>publishName</i>	Publish name or empty if publishId should be used instead
<i>password</i>	Required password or an empty string if no password
<i>subscribeName</i>	Name of created favorite (usually same as the publish)
<i>subscribeDelegate</i>	Gets notified of the result

12.134.3.4 subscribe() [3/3]

```
void cbe::SubscribeManager::subscribe (
    std::string sharingUserName,
    std::string publishName,
    SubscribeDelegatePtr subscribeDelegate ) [inline]
```

Subscribes to a [Publish](#) using names.

Pointer to [cbe::delegate::SubscribeDelegate](#) that is passed into asynchronous version of method [subscribe\(\)](#)

Overload of [subscribe\(\)](#)

Parameters

<i>sharingUserName</i>	User name of the owner of the publish
<i>publishName</i>	Publish name that will also be the name of created favorite.
<i>subscribeDelegate</i>	Gets notified of the result

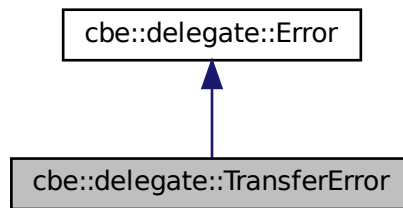
The documentation for this class was generated from the following file:

- include/cbe/SubscribeManager.h

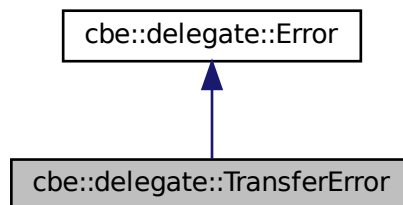
12.135 cbe::delegate::TransferError Class Reference

```
#include <TransferError.h>
```

Inheritance diagram for `cbe::delegate::TransferError`:



Collaboration diagram for `cbe::delegate::TransferError`:



Public Member Functions

- **TransferError** ([Error](#) &&error, std::string name, [cbe::ObjectId](#) objectId, [cbe::ContainerId](#) parentId)

Public Attributes

- std::string **name** {}
- [cbe::ObjectId](#) **objectId** {}
- [cbe::ContainerId](#) **parentId** {}

Friends

- std::ostream & **operator**<< (std::ostream &os, const [TransferError](#) &transferError)

12.135.1 Detailed Description

Contains error information delivered from CloudBackend SDK in connection with a failed transfer, i.e., upload/download.

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

- include/cbe/delegate/TransferError.h

12.136 cbe::delegate::UnBanDelegate Class Reference

```
#include <UnBanDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [UnBanSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onUnBanSuccess](#) (std::string &&memberName, [cbe::MemberId](#) memberId)=0
- virtual void [onUnBanError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.136.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Member::unBan](#)

12.136.2 Member Function Documentation

12.136.2.1 onUnBanError()

```
virtual void cbe::delegate::UnBanDelegate::onUnBanError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.136.2.2 onUnBanSuccess()

```
virtual void cbe::delegate::UnBanDelegate::onUnBanSuccess (
    std::string && memberName,
    cbe::MemberId memberId ) [pure virtual]
```

Called upon successful unBan.

Parameters

<i>memberName</i>	Name of the member being unbanned.
<i>memberId</i>	Id of the member being unbanned.

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

- include/cbe/delegate/UnBanDelegate.h

12.137 cbe::delegate::UnBanSuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::UnBanDelegate::onUnBanSuccess](#).
`#include <UnBanDelegate.h>`

Public Member Functions

- **UnBanSuccess** ([cbe::DefaultCtor](#))
- **UnBanSuccess** (std::string &&memberName, [cbe::MemberId](#) memberId)

Public Attributes

- `std::string` **memberName** {}
- `cbe::MemberId` **memberId** {}

12.137.1 Detailed Description

Convenience type that bundles all parameters passed to method `cbe::delegate::UnBanDelegate::onUnBanSuccess`. The documentation for this class was generated from the following file:

- `include/cbe/delegate/UnBanDelegate.h`

12.138 cbe::delegate::UnPublishDelegate Class Reference

```
#include <UnPublishDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [UnPublishSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onUnPublishSuccess](#) ([cbe::PublishId](#) publishId, [cbe::ItemId](#) itemId)=0
- virtual void [onUnPublishError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.138.1 Detailed Description

Delegate class for the asynchronous version of methods:

- [cbe::Container::unPublish](#)
- [cbe::Object::unPublish](#)

12.138.2 Member Function Documentation

12.138.2.1 onUnPublishError()

```
virtual void cbe::delegate::UnPublishDelegate::onUnPublishError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error was encountered.

12.138.2.2 onUnPublishSuccess()

```
virtual void cbe::delegate::UnPublishDelegate::onUnPublishSuccess (
    cbe::PublishId publishId,
    cbe::ItemId itemId ) [pure virtual]
```

Called upon successful unPublish.

Parameters

<i>publishId</i>	Id of the publication being published.
<i>itemId</i>	Id of the item - cbe::Object or cbe::Container - that has been unPublished

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

- include/cbe/delegate/UnPublishDelegate.h

12.139 cbe::delegate::UnPublishSuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::UnPublishDelegate::onUnPublishError](#).

```
#include <UnPublishDelegate.h>
```

Public Member Functions

- **UnPublishSuccess** ([cbe::DefaultCtor](#))
- **UnPublishSuccess** ([cbe::PublishId](#) publishId, [cbe::ItemId](#) itemId)
- [operator bool](#) () const

Checks if current instance is valid.

Public Attributes

- [cbe::PublishId](#) publishId {}
- [cbe::ItemId](#) itemId {}

12.139.1 Detailed Description

Convenience type that bundles all parameters passed to method [cbe::delegate::UnPublishDelegate::onUnPublishError](#).

12.139.2 Member Function Documentation

12.139.2.1 operator bool()

```
cbe::delegate::UnPublishSuccess::operator bool ( ) const [explicit]
```

Checks if current instance is valid.

Returns

`true`: is valid

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

- include/cbe/delegate/UnPublishDelegate.h

12.140 cbe::delegate::UnShareDelegate Class Reference

```
#include <UnShareDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [UnShareSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onUnShareSuccess](#) (std::string &&message)=0
- virtual void [onUnShareError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.140.1 Detailed Description

Delegate class for the asynchronous version of methods:

- [cbe::Container::unShare\(\)](#)
- [cbe::Object::unShare\(\)](#)

12.140.2 Member Function Documentation

12.140.2.1 onUnShareError()

```
virtual void cbe::delegate::UnShareDelegate::onUnShareError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.140.2.2 onUnShareSuccess()

```
virtual void cbe::delegate::UnShareDelegate::onUnShareSuccess (
    std::string && message ) [pure virtual]
```

Called upon successful UnShare.

Parameters

<i>message</i>	Message of the Unshare.
----------------	-------------------------

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

- include/cbe/delegate/UnShareDelegate.h

12.141 cbe::delegate::UnShareSuccess Class Reference

Convenience type that bundles the parameter passed to method [cbe::delegate::ShareDelegate::onShareSuccess](#).

```
#include <UnShareDelegate.h>
```

Public Member Functions

- **UnShareSuccess** ([cbe::DefaultCtor](#))
- **UnShareSuccess** (std::string &&message)
- [operator bool](#) () const

Checks if current instance is valid.

Public Attributes

- std::string **message** {}

12.141.1 Detailed Description

Convenience type that bundles the parameter passed to method [cbe::delegate::ShareDelegate::onShareSuccess](#).

12.141.2 Member Function Documentation

12.141.2.1 operator bool()

`cbe::delegate::UnShareSuccess::operator bool ( ) const [explicit]`
 Checks if current instance is valid.

Returns

`true`: is valid

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

- `include/cbe/delegate/UnShareDelegate.h`

12.142 cbe::delegate::UnSubscribeDelegate Class Reference

```
#include <UnSubscribeDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [UnSubscribeSuccess](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onUnSubscribeSuccess](#) ([cbe::PublishId](#) publishId, [cbe::ItemId](#) itemId)=0
- virtual void [onUnSubscribeError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.142.1 Detailed Description

Delegate class for the asynchronous version of methods:

- [cbe::Container::unSubscribe](#)
- [cbe::Object::unSubscribe](#)

12.142.2 Member Function Documentation

12.142.2.1 onUnSubscribeError()

```
virtual void cbe::delegate::UnSubscribeDelegate::onUnSubscribeError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.142.2.2 onUnSubscribeSuccess()

```
virtual void cbe::delegate::UnSubscribeDelegate::onUnSubscribeSuccess (
    cbe::PublishId publishId,
    cbe::ItemId itemId ) [pure virtual]
```

Called upon successful UnSubscribe.

Parameters

<i>publishId</i>	The PublishId of the UnSubscribe.
<i>itemId</i>	Instance of ItemId that is being UnSubscribed.

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

- include/cbe/delegate/UnSubscribeDelegate.h

12.143 cbe::delegate::UnSubscribeSuccess Class Reference

Convenience type that bundles all parameters passed to method [cbe::delegate::UnSubscribeDelegate::onUnSubscribeSuccess](#).

```
#include <UnSubscribeDelegate.h>
```

Public Member Functions

- **UnSubscribeSuccess** ([cbe::PublishId](#) publishId, [cbe::ItemId](#) itemId)
- [operator bool](#) () const
Checks if current instance is valid.

Public Attributes

- [cbe::PublishId](#) publishId {}
- [cbe::ItemId](#) itemId {}

12.143.1 Detailed Description

Convenience type that bundles all parameters passed to method [cbe::delegate::UnSubscribeDelegate::onUnSubscribeSuccess](#).

12.143.2 Member Function Documentation

12.143.2.1 operator bool()

```
cbe::delegate::UnSubscribeSuccess::operator bool ( ) const [explicit]
```

Checks if current instance is valid.

Returns

true: is valid

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

- include/cbe/delegate/UnSubscribeDelegate.h

12.144 cbe::delegate::UpdateKeyValuesDelegate Class Reference

```
#include <UpdateKeyValuesDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Object](#)
- using **Error** = [delegate::Error](#)

Public Member Functions

- virtual void [onUpdateKeyValuesSuccess](#) ([cbe::Object](#) &&object)=0
- virtual void [onUpdateKeyValuesError](#) ([Error](#) &&error, [cbe::util::Context](#) &&context)=0

12.144.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Object::updateKeyValues\(\)](#)

12.144.2 Member Function Documentation

12.144.2.1 onUpdateKeyValuesError()

```
virtual void cbe::delegate::UpdateKeyValuesDelegate::onUpdateKeyValuesError (
    Error && error,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.144.2.2 onUpdateKeyValuesSuccess()

```
virtual void cbe::delegate::UpdateKeyValuesDelegate::onUpdateKeyValuesSuccess (
    cbe::Object && object ) [pure virtual]
```

Called upon successful UpdateKeyValues.

Parameters

<i>object</i>	Instance of object that is getting its Key/Values updated.
---------------	--

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

- include/cbe/delegate/UpdateKeyValuesDelegate.h

12.145 cbe::delegate::UploadDelegate Class Reference

```
#include <UploadDelegate.h>
```

Classes

- struct [ErrorInfo](#)

Public Types

- using **Success** = [cbe::Object](#)
- using **Error** = [TransferError](#)

Public Member Functions

- virtual void [onUploadSuccess](#) ([cbe::Object](#) &&object)=0
- virtual void [onUploadError](#) ([cbe::delegate::TransferError](#) &&transferError, [cbe::util::Context](#) &&context)=0
- virtual void [onChunkSent](#) ([cbe::Object](#) &&object, std::uint64_t sent, std::uint64_t total)

12.145.1 Detailed Description

Delegate class for the asynchronous version of method:

- [cbe::Container::upload\(const std::string&, UploadDelegatePtr\)](#)
See [Async example](#) of creation of a [cbe::Object](#) by using [Container::upload\(\)](#)
- [cbe::Container::upload\(const std::string&, const std::string&, UploadDelegatePtr\)](#)
- [cbe::Object::uploadStream\(const std::string&, cbe::StreamId, UploadDelegatePtr\)](#)

12.145.2 Member Function Documentation

12.145.2.1 onChunkSent()

```
virtual void cbe::delegate::UploadDelegate::onChunkSent (
    cbe::Object && object,
    std::uint64_t sent,
    std::uint64_t total ) [virtual]
```

Called when a chunk of data has been sent.

Parameters

<i>object</i>	Object associated with current upload.
<i>sent</i>	Size, in number of bytes, of the sent chunk.
<i>total</i>	Total number of bytes sent so far.

12.145.2.2 onUploadError()

```
virtual void cbe::delegate::UploadDelegate::onUploadError (
    cbe::delegate::TransferError && transferError,
    cbe::util::Context && context ) [pure virtual]
```

Called if an error is encountered.

12.145.2.3 onUploadSuccess()

```
virtual void cbe::delegate::UploadDelegate::onUploadSuccess (
    cbe::Object && object ) [pure virtual]
```

Called upon successful Upload.

Parameters

<i>object</i>	Instance of object that is being uploaded.
---------------	--

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

- include/cbe/delegate/UploadDelegate.h

Index

Account
 cbe::Account, [52](#)
account
 cbe::CloudBackend, [61](#)
AccountStatus
 cbe, [37](#)
AclDelegatePtr
 cbe::delegate, [42](#)
AclGroupId
 cbe, [36](#)
AclMap
 cbe, [36](#)
aclMap
 cbe::Item, [167](#)
AclScope
 cbe, [37](#)
aclTag
 cbe::Item, [167](#)
addListener
 cbe::CloudBackend, [61](#)
addRoleMember
 cbe::Role, [228](#)
AddRoleMemberDelegatePtr
 cbe::delegate, [43](#)
 cbe::Role, [227](#)

ban
 cbe::Member, [183](#)
BanDelegatePtr
 cbe::delegate, [43](#)
 cbe::Member, [182](#)

castContainer
 cbe::CloudBackend, [61](#)
castObject
 cbe::CloudBackend, [62](#)
cbe, [33](#)
 AccountStatus, [37](#)
 AclGroupId, [36](#)
 AclMap, [36](#)
 AclScope, [37](#)
 Date, [36](#)
 DefaultCtor, [37](#)
 ErrorCode, [36](#)
 FilterOrder, [38](#)
 ItemType, [38](#)
 KeyValues, [36](#)
 MemberBanInfo, [37](#)
 ObjectType, [38](#)
 Permissions, [38](#)
 PublishAccess, [39](#)
 PublishVisibility, [39](#)
 Visibility, [40](#)
cbe::Account, [51](#)
 Account, [52](#)
 client, [52](#)
 databases, [52](#)
 firstName, [52](#)
 libContainerId, [52](#)
 operator bool, [52](#)
 password, [52](#)
 rootContainer, [53](#)
 rootDatabase, [53](#)
 source, [53](#)
 tenantContainerId, [53](#)
 username, [53](#)
cbe::CloudBackend, [58](#)
 account, [61](#)
 addListener, [61](#)
 castContainer, [61](#)
 castObject, [62](#)
 clearCache, [62](#)
 CloudBackend, [61](#)
 CloudBackendListenerDelegatePtr, [60](#)
 createAccount, [62](#)
 CreateAccountDelegatePtr, [60](#)
 groupManager, [62](#)
 login, [63](#)
 LoginDelegatePtr, [60](#)
 operator bool, [64](#)
 publishManager, [64](#)
 query, [65](#), [66](#)
 QueryDelegatePtr, [60](#)
 QueryJoinDelegatePtr, [60](#)
 queryWithPath, [66](#), [67](#)
 removeListener, [67](#)
 search, [67](#), [68](#)
 shareManager, [68](#)
 subscribeManager, [68](#)
 terminate, [68](#)
 version, [69](#)
cbe::Container, [72](#)
 createContainer, [75](#)
 CreateContainerDelegatePtr, [74](#)
 createObject, [76](#)
 CreateObjectDelegatePtr, [74](#)
 getAcl, [76](#)
 GetAclDelegatePtr, [74](#)
 move, [77](#)

- MoveDelegatePtr, 74
- publish, 77
- PublishDelegatePtr, 74
- query, 77, 78
- QueryDelegatePtr, 74
- QueryJoinDelegatePtr, 75
- queryWithPath, 78
- remove, 78
- RemoveDelegatePtr, 75
- rename, 79
- RenameDelegatePtr, 75
- search, 79
- setAcl, 80
- SetAclDelegatePtr, 75
- share, 80
- ShareDelegatePtr, 75
- unPublish, 80
- UnPublishDelegatePtr, 75
- unShare, 81
- UnShareDelegatePtr, 75
- unsubscribe, 81
- UnSubscribeDelegatePtr, 75
- upload, 81–83
- UploadDelegatePtr, 75
- cbe::Database, 88
 - containerId, 88
 - created, 88
 - databaseId, 89
 - name, 89
 - ownerId, 89
 - readLocked, 89
 - rootContainer, 89
 - writeLocked, 89
- cbe::delegate, 40
 - AclDelegatePtr, 42
 - AddRoleMemberDelegatePtr, 43
 - BanDelegatePtr, 43
 - CloudBackendListenerDelegatePtr, 43
 - CreateAccountDelegatePtr, 43
 - CreateContainerDelegatePtr, 43
 - CreateGroupDelegatePtr, 43
 - CreateObjectDelegatePtr, 43
 - CreateRoleDelegatePtr, 43
 - DownloadBinaryDelegatePtr, 44
 - DownloadDelegatePtr, 44
 - GetStreamsDelegatePtr, 44
 - GetSubscriptionsDelegatePtr, 44
 - ICloudBackendListenerPtr, 44
 - JoinDelegatePtr, 44
 - KickDelegatePtr, 44
 - LeaveDelegatePtr, 44
 - ListGroupsDelegatePtr, 45
 - ListMembersDelegatePtr, 45
 - ListRolesDelegatePtr, 45
 - ListSharesDelegatePtr, 45
 - LogInDelegatePtr, 45
 - ProgressEventFn, 45
 - PublishDelegatePtr, 45
 - QueryDelegatePtr, 45
 - QueryJoinDelegatePtr, 46
 - RemoveRoleDelegatePtr, 46
 - RemoveRoleMemberDelegatePtr, 46
 - SearchGroupsDelegatePtr, 46
 - ShareDelegatePtr, 46
 - SubscribeDelegatePtr, 46
 - UnBanDelegatePtr, 46
 - UnPublishDelegatePtr, 47
 - UnShareDelegatePtr, 47
 - UnSubscribeDelegatePtr, 47
 - UpdateKeyValuesDelegatePtr, 47
 - UploadDelegatePtr, 47
 - cbe::delegate::AclDelegate, 53
 - onAclError, 54
 - onAclSuccess, 54
 - cbe::delegate::AclDelegate::ErrorInfo, 98
 - cbe::delegate::AddRoleMemberDelegate, 54
 - onAddRoleMemberError, 55
 - onAddRoleMemberSuccess, 55
 - cbe::delegate::AddRoleMemberDelegate::ErrorInfo, 99
 - cbe::delegate::BanDelegate, 56
 - onBanError, 56
 - onBanSuccess, 57
 - cbe::delegate::BanDelegate::ErrorInfo, 100
 - cbe::delegate::BanSuccess, 57
 - cbe::delegate::ChunkTransferred, 57
 - cbe::delegate::CloudBackendListenerDelegate, 69
 - onRemoteContainerAdded, 70
 - onRemoteContainerMoved, 70
 - onRemoteContainerRemoved, 70
 - onRemoteContainerRenamed, 70
 - onRemoteObjectAdded, 71
 - onRemoteObjectMoved, 71
 - onRemoteObjectRemoved, 71
 - onRemoteObjectRenamed, 71
 - cbe::delegate::container, 47
 - MoveDelegatePtr, 48
 - RemoveDelegatePtr, 48
 - RenameDelegatePtr, 48
 - cbe::delegate::container::MoveDelegate, 184
 - onMoveError, 185
 - onMoveSuccess, 185
 - cbe::delegate::container::MoveDelegate::ErrorInfo, 102
 - cbe::delegate::container::RemoveDelegate, 217
 - onRemoveError, 218
 - onRemoveSuccess, 218
 - cbe::delegate::container::RemoveDelegate::ErrorInfo, 103
 - cbe::delegate::container::RemoveSuccess, 222
 - cbe::delegate::container::RenameDelegate, 223
 - onRenameError, 223
 - onRenameSuccess, 223
 - cbe::delegate::container::RenameDelegate::ErrorInfo, 104
 - cbe::delegate::CreateAccountDelegate, 83
 - onCreateAccountError, 84
 - onCreateAccountSuccess, 84

- cbe::delegate::CreateAccountDelegate::ErrorInfo, 105
- cbe::delegate::CreateContainerDelegate, 84
 - onCreateContainerError, 85
 - onCreateContainerSuccess, 85
- cbe::delegate::CreateContainerDelegate::ErrorInfo, 106
- cbe::delegate::CreateGroupDelegate, 85
 - onCreateGroupError, 86
 - onCreateGroupSuccess, 86
- cbe::delegate::CreateGroupDelegate::ErrorInfo, 107
- cbe::delegate::CreateObjectDelegate, 86
 - onCreateObjectError, 86
 - onCreateObjectSuccess, 86
- cbe::delegate::CreateObjectDelegate::ErrorInfo, 108
- cbe::delegate::CreateRoleDelegate, 87
 - onCreateRoleError, 87
 - onCreateRoleSuccess, 87
- cbe::delegate::CreateRoleDelegate::ErrorInfo, 109
- cbe::delegate::DownloadBinaryDelegate, 90
 - onChunkReceived, 90
 - onDownloadBinaryError, 90
 - onDownloadBinarySuccess, 91
- cbe::delegate::DownloadBinaryDelegate::ErrorInfo, 110
- cbe::delegate::DownloadBinarySuccess, 91
 - operator bool, 92
- cbe::delegate::DownloadDelegate, 92
 - onChunkReceived, 92
 - onDownloadError, 93
 - onDownloadSuccess, 93
- cbe::delegate::DownloadDelegate::ErrorInfo, 111
- cbe::delegate::DownloadSuccess, 93
 - operator bool, 94
- cbe::delegate::Error, 95
 - errorCode, 96
 - message, 96
 - operator bool, 95
 - operator<<, 95
 - reason, 96
- cbe::delegate::GetStreamsDelegate, 150
 - onGetStreamsError, 150
 - onGetStreamsSuccess, 150
- cbe::delegate::GetStreamsDelegate::ErrorInfo, 112
- cbe::delegate::GetSubscriptionsDelegate, 151
 - onGetSubscriptionsError, 151
 - onGetSubscriptionsSuccess, 151
- cbe::delegate::GetSubscriptionsDelegate::ErrorInfo, 113
- cbe::delegate::group, 48
 - JoinDelegatePtr, 49
 - RemoveDelegatePtr, 49
 - RenameDelegatePtr, 49
- cbe::delegate::group::JoinDelegate, 171
 - onJoinError, 171
 - onJoinSuccess, 171
- cbe::delegate::group::JoinDelegate::ErrorInfo, 114
- cbe::delegate::group::RemoveDelegate, 218
 - onRemoveError, 219
 - onRemoveSuccess, 219
- cbe::delegate::group::RemoveDelegate::ErrorInfo, 115
- cbe::delegate::group::RemoveSuccess, 222
- cbe::delegate::group::RenameDelegate, 224
 - onRenameError, 224
 - onRenameSuccess, 224
- cbe::delegate::group::RenameDelegate::ErrorInfo, 116
- cbe::delegate::group::RenameSuccess, 225
 - operator bool, 226
- cbe::delegate::ICloudBackendListener, 164
 - onRemoteContainerAdded, 164
 - onRemoteContainerMoved, 164
 - onRemoteContainerRemoved, 165
 - onRemoteContainerRenamed, 165
 - onRemoteObjectAdded, 165
 - onRemoteObjectMoved, 165
 - onRemoteObjectRemoved, 165
 - onRemoteObjectRenamed, 166
- cbe::delegate::ItemError, 170
 - onItemError, 170
- cbe::delegate::ItemError::Error, 96
- cbe::delegate::JoinDelegate, 171
 - ErrorInfo, 172
 - onJoinError, 172
 - onJoinSuccess, 172
- cbe::delegate::JoinError, 172
 - onJoinError, 173
- cbe::delegate::JoinError::Error, 97
- cbe::delegate::KickDelegate, 173
 - onKickError, 174
 - onKickSuccess, 174
- cbe::delegate::KickDelegate::ErrorInfo, 117
- cbe::delegate::KickSuccess, 174
- cbe::delegate::LeaveDelegate, 174
 - onLeaveError, 175
 - onLeaveSuccess, 175
- cbe::delegate::LeaveDelegate::ErrorInfo, 118
- cbe::delegate::LeaveSuccess, 175
 - operator bool, 175
- cbe::delegate::ListGroupDelegate, 176
 - onListGroupError, 176
 - onListGroupSuccess, 176
- cbe::delegate::ListGroupDelegate::ErrorInfo, 119
- cbe::delegate::ListMembersDelegate, 177
 - onListMembersError, 177
 - onListMembersSuccess, 177
- cbe::delegate::ListMembersDelegate::ErrorInfo, 120
- cbe::delegate::ListRolesDelegate, 177
 - onListRolesError, 178
 - onListRolesSuccess, 178
- cbe::delegate::ListRolesDelegate::ErrorInfo, 121
- cbe::delegate::ListSharesDelegate, 178
 - onListSharesError, 179
 - onListSharesSuccess, 179
- cbe::delegate::ListSharesDelegate::ErrorInfo, 122
- cbe::delegate::LoadError, 179
 - onLoadError, 179
- cbe::delegate::LoadError::Error, 97
 - errorCode, 97
- cbe::delegate::LogInDelegate, 180

- onLogInError, 180
- onLogInSuccess, 181
- Success, 180
- cbe::delegate::LogInDelegate::ErrorInfo, 123
- cbe::delegate::object, 49
 - MoveDelegatePtr, 50
 - RemoveDelegatePtr, 50
 - RenameDelegatePtr, 50
- cbe::delegate::object::MoveDelegate, 185
 - onMoveError, 185
 - onMoveSuccess, 185
- cbe::delegate::object::MoveDelegate::ErrorInfo, 124
- cbe::delegate::object::RemoveDelegate, 219
 - onRemoveError, 219
 - onRemoveSuccess, 220
- cbe::delegate::object::RemoveDelegate::ErrorInfo, 125
- cbe::delegate::object::RemoveSuccess, 222
 - operator bool, 223
- cbe::delegate::object::RenameDelegate, 225
 - onRenameError, 225
 - onRenameSuccess, 225
- cbe::delegate::object::RenameDelegate::ErrorInfo, 126
- cbe::delegate::PublishDelegate, 202
 - onPublishError, 202
 - onPublishSuccess, 202
- cbe::delegate::PublishDelegate::ErrorInfo, 127
- cbe::delegate::PublishSuccess, 204
 - operator bool, 204
- cbe::delegate::QueryDelegate, 211
 - onQueryError, 212
 - onQuerySuccess, 212
- cbe::delegate::QueryDelegate::ErrorInfo, 128
- cbe::delegate::QueryError, 212
- cbe::delegate::QueryJoinDelegate, 213
 - ErrorInfo, 214
 - onQueryJoinError, 214
 - onQueryJoinSuccess, 214
- cbe::delegate::QueryLoaded, 215
 - onQuerySuccess, 215
- cbe::delegate::RemoveRoleDelegate, 220
 - onRemoveRoleError, 220
 - onRemoveRoleSuccess, 220
- cbe::delegate::RemoveRoleDelegate::ErrorInfo, 129
- cbe::delegate::RemoveRoleMemberDelegate, 221
 - onRemoveRoleMemberError, 221
 - onRemoveRoleMemberSuccess, 221
- cbe::delegate::RemoveRoleMemberDelegate::ErrorInfo, 130
- cbe::delegate::SearchGroupsDelegate, 229
 - onSearchGroupsError, 230
 - onSearchGroupsSuccess, 230
- cbe::delegate::SearchGroupsDelegate::ErrorInfo, 131
- cbe::delegate::ShareDelegate, 231
 - onShareError, 231
 - onShareSuccess, 231
- cbe::delegate::ShareDelegate::ErrorInfo, 132
- cbe::delegate::ShareSuccess, 233
 - operator bool, 234
- cbe::delegate::SubscribeDelegate, 236
 - onSubscribeError, 236
 - onSubscribeSuccess, 237
- cbe::delegate::SubscribeDelegate::ErrorInfo, 133
- cbe::delegate::TransferError, 239
- cbe::delegate::UnBanDelegate, 240
 - onUnBanError, 241
 - onUnBanSuccess, 241
- cbe::delegate::UnBanDelegate::ErrorInfo, 134
- cbe::delegate::UnBanSuccess, 241
- cbe::delegate::UnPublishDelegate, 242
 - onUnPublishError, 242
 - onUnPublishSuccess, 242
- cbe::delegate::UnPublishDelegate::ErrorInfo, 135
- cbe::delegate::UnPublishSuccess, 243
 - operator bool, 243
- cbe::delegate::UnShareDelegate, 243
 - onUnShareError, 244
 - onUnShareSuccess, 244
- cbe::delegate::UnShareDelegate::ErrorInfo, 136
- cbe::delegate::UnShareSuccess, 244
 - operator bool, 244
- cbe::delegate::UnSubscribeDelegate, 245
 - onUnSubscribeError, 245
 - onUnSubscribeSuccess, 245
- cbe::delegate::UnSubscribeDelegate::ErrorInfo, 137
- cbe::delegate::UnSubscribeSuccess, 246
 - operator bool, 246
- cbe::delegate::UpdateKeyValuesDelegate, 246
 - onUpdateKeyValuesError, 247
 - onUpdateKeyValuesSuccess, 247
- cbe::delegate::UpdateKeyValuesDelegate::ErrorInfo, 138
- cbe::delegate::UploadDelegate, 247
 - onChunkSent, 248
 - onUploadError, 248
 - onUploadSuccess, 248
- cbe::delegate::UploadDelegate::ErrorInfo, 139
- cbe::Filter, 145
 - getDataType, 147
 - getItemOrder, 147
 - getOffset, 147
 - getQuery, 147
 - setAscending, 147
 - setByPassCache, 148
 - setContainerFirst, 148
 - setCount, 148
 - setDataType, 148
 - setDeleted, 148
 - setItemOrder, 149
 - setOffset, 149
 - setQuery, 149
- cbe::Group, 151
 - createGroup, 154
 - CreateGroupDelegatePtr, 153
 - createRole, 154
 - CreateRoleDelegatePtr, 153
 - getVisibility, 155

- Group, 154
- groupContainer, 155
- id, 155
- join, 155
- JoinDelegatePtr, 153
- joined, 155
- leave, 155
- LeaveDelegatePtr, 153
- listBannedMembers, 156
- ListBannedMembersDelegatePtr, 153
- listMembers, 156
- ListMembersDelegatePtr, 153
- listRoles, 156
- ListRolesDelegatePtr, 153
- name, 156
- operator bool, 156
- parentId, 157
- remove, 157
- RemoveDelegatePtr, 153
- removeRole, 157
- RemoveRoleDelegatePtr, 153
- rename, 157
- RenameDelegatePtr, 153
- requests, 157
- cbe::GroupFilter, 158
 - getAscending, 158
 - getCount, 158
 - getDeleted, 158
 - getFilter, 159
 - getGroupOrder, 159
 - getOffset, 159
 - getOrder, 159
 - getPublicFirst, 159
 - getQuery, 159
 - operator<<, 160
 - setAscending, 159
 - setCount, 159
 - setDeleted, 159
 - setFilter, 159
 - setOffset, 159
 - setQuery, 160
- cbe::GroupManager, 160
 - getTenantId, 161
 - GroupManager, 161
 - listGroups, 161
 - ListGroupsDelegatePtr, 161
 - operator bool, 161
 - searchGroups, 162
 - SearchGroupsDelegatePtr, 161
- cbe::GroupQueryResult, 162
 - containsGroup, 162
 - filter, 163
 - getGroupsSnapshot, 163
 - GroupsLoaded, 163
 - totalCount, 163
- cbe::Item, 166
 - aclMap, 167
 - aclTag, 167
 - created, 168
 - dataLoaded, 168
 - deleted, 168
 - description, 168
 - driveId, 168
 - getPublished, 168
 - getShareFromUserId, 168
 - getShareIds, 168
 - getSubscribe, 168
 - getUserFromShareId, 168
 - hasPublished, 169
 - hasSubscribe, 169
 - id, 169
 - idLoaded, 169
 - name, 169
 - oldParentId, 169
 - operator bool, 169
 - ownerId, 169
 - parentId, 169
 - path, 169
 - type, 170
 - updated, 170
 - username, 170
- cbe::MatchesID< IdT >, 181
- cbe::Member, 181
 - ban, 183
 - BanDelegatePtr, 182
 - groupId, 183
 - kick, 183
 - KickDelegatePtr, 182
 - Member, 183
 - memberId, 183
 - name, 183
 - operator bool, 184
 - unBan, 184
 - UnBanDelegatePtr, 182
 - visibility, 184
- cbe::Object, 186
 - download, 190
 - DownloadBinaryDelegatePtr, 188
 - DownloadDelegatePtr, 188
 - downloadStream, 190
 - getAcl, 191
 - GetAclDelegatePtr, 188
 - getMimeType, 191
 - getObjectType, 192
 - getStreams, 192
 - GetStreamsDelegatePtr, 188
 - move, 193
 - MoveDelegatePtr, 188
 - publish, 193
 - PublishDelegatePtr, 189
 - remove, 193
 - RemoveDelegatePtr, 189
 - rename, 194
 - RenameDelegatePtr, 189
 - setAcl, 194
 - SetAclDelegatePtr, 189

- share, 194
- ShareDelegatePtr, 189
- Streams, 189
- unPublish, 195
- UnPublishDelegatePtr, 189
- unShare, 195
- UnShareDelegatePtr, 189
- unSubscribe, 195
- UnSubscribeDelegatePtr, 189
- updateKeyValues, 195, 196
- UpdateKeyValuesDelegatePtr, 189
- UploadDelegatePtr, 189
- uploadStream, 196
- cbe::Publish, 199
 - getDescription, 200
 - getPassword, 200
 - getPrivacy, 200
 - getPublishId, 200
 - getSecurity, 200
 - getTitle, 200
 - operator bool, 200
 - Publish, 200
 - setDescription, 201
 - setPassword, 201
 - setPrivacy, 201
 - setSecurity, 201
 - setTitle, 201
- cbe::PublishManager, 203
 - getPublished, 203
 - operator bool, 203
 - PublishManager, 203
- cbe::QueryChain, 204
 - getQueryResult, 205
 - join, 206
- cbe::QueryChain::Exception, 142
- cbe::QueryChainExt, 207
 - join, 208–210
- cbe::QueryResult, 215
 - containersLoaded, 216
 - containsItem, 216
 - filter, 216
 - getItemsSnapshot, 216
 - itemsLoaded, 217
 - objectsLoaded, 217
 - operator bool, 217
 - QueryResult, 216
 - totalCount, 217
- cbe::Request, 226
- cbe::Role, 227
 - addRoleMember, 228
 - AddRoleMemberDelegatePtr, 227
 - groupId, 228
 - id, 228
 - listMembers, 229
 - ListMembersDelegatePtr, 228
 - name, 229
 - operator bool, 229
 - removeRoleMember, 229
 - RemoveRoleMemberDelegatePtr, 228
 - Role, 228
- cbe::ShareData, 230
- cbe::ShareManager, 232
 - listAvailableShares, 232
 - listMyShares, 233
 - ListSharesDelegatePtr, 232
 - operator bool, 233
 - ShareManager, 232
- cbe::Stream, 234
 - length, 234
 - streamId, 234
- cbe::Subscribe, 234
 - getDate, 235
 - getDescription, 235
 - getOwner, 235
 - getPassword, 235
 - getPrivacy, 235
 - getPublishId, 235
 - getSecurity, 236
 - getSubscribed, 236
 - getTitle, 236
 - unsubscribe, 236
- cbe::SubscribeManager, 237
 - getSubscriptions, 238
 - GetSubscriptionsDelegatePtr, 237
 - subscribe, 238, 239
 - SubscribeDelegatePtr, 238
- cbe::util, 50
- cbe::util::Context, 83
- cbe::util::ErrorInfo, 140
- cbe::util::ErrorInfoImpl< ErrorT >, 141
- cbe::util::Exception, 143
 - derivedCbeException, 144
- cbe::util::ExceptionImpl< ErrorInfoT >, 144
- cbe::util::Optional< T >, 197
 - operator bool, 198
- cbe::util::Optional< T >::BadAccess, 55
- clearCache
 - cbe::CloudBackend, 62
- client
 - cbe::Account, 52
- CloudBackend
 - cbe::CloudBackend, 61
- CloudBackendListenerDelegatePtr
 - cbe::CloudBackend, 60
 - cbe::delegate, 43
- containerId
 - cbe::Database, 88
- containersLoaded
 - cbe::QueryResult, 216
- containsGroup
 - cbe::GroupQueryResult, 162
- containsItem
 - cbe::QueryResult, 216
- createAccount
 - cbe::CloudBackend, 62
- CreateAccountDelegatePtr

- cbe::CloudBackend, 60
- cbe::delegate, 43
- createContainer
 - cbe::Container, 75
- CreateContainerDelegatePtr
 - cbe::Container, 74
 - cbe::delegate, 43
- created
 - cbe::Database, 88
 - cbe::Item, 168
- createGroup
 - cbe::Group, 154
- CreateGroupDelegatePtr
 - cbe::delegate, 43
 - cbe::Group, 153
- createObject
 - cbe::Container, 76
- CreateObjectDelegatePtr
 - cbe::Container, 74
 - cbe::delegate, 43
- createRole
 - cbe::Group, 154
- CreateRoleDelegatePtr
 - cbe::delegate, 43
 - cbe::Group, 153
- databaseld
 - cbe::Database, 89
- databases
 - cbe::Account, 52
- dataLoaded
 - cbe::Item, 168
- Date
 - cbe, 36
- DefaultCtor
 - cbe, 37
- deleted
 - cbe::Item, 168
- derivedCbeException
 - cbe::util::Exception, 144
- description
 - cbe::Item, 168
- download
 - cbe::Object, 190
- DownloadBinaryDelegatePtr
 - cbe::delegate, 44
 - cbe::Object, 188
- DownloadDelegatePtr
 - cbe::delegate, 44
 - cbe::Object, 188
- downloadStream
 - cbe::Object, 190
- driveld
 - cbe::Item, 168
- ErrorCode
 - cbe, 36
- errorCode
 - cbe::delegate::Error, 96
- cbe::delegate::LoadError::Error, 97
- ErrorInfo
 - cbe::delegate::JoinDelegate, 172
 - cbe::delegate::QueryJoinDelegate, 214
- filter
 - cbe::GroupQueryResult, 163
 - cbe::QueryResult, 216
- FilterOrder
 - cbe, 38
- firstName
 - cbe::Account, 52
- getAcl
 - cbe::Container, 76
 - cbe::Object, 191
- GetAclDelegatePtr
 - cbe::Container, 74
 - cbe::Object, 188
- getAscending
 - cbe::GroupFilter, 158
- getCount
 - cbe::GroupFilter, 158
- getDataType
 - cbe::Filter, 147
- getDate
 - cbe::Subscribe, 235
- getDeleted
 - cbe::GroupFilter, 158
- getDescription
 - cbe::Publish, 200
 - cbe::Subscribe, 235
- getFilter
 - cbe::GroupFilter, 159
- getGroupOrder
 - cbe::GroupFilter, 159
- getGroupsSnapshot
 - cbe::GroupQueryResult, 163
- getItemOrder
 - cbe::Filter, 147
- getItemsSnapshot
 - cbe::QueryResult, 216
- getMimeType
 - cbe::Object, 191
- getObjectType
 - cbe::Object, 192
- getOffset
 - cbe::Filter, 147
 - cbe::GroupFilter, 159
- getOrder
 - cbe::GroupFilter, 159
- getOwner
 - cbe::Subscribe, 235
- getPassword
 - cbe::Publish, 200
 - cbe::Subscribe, 235
- getPrivacy
 - cbe::Publish, 200
 - cbe::Subscribe, 235

- getPublicFirst
 - cbe::GroupFilter, 159
- getPublished
 - cbe::Item, 168
 - cbe::PublishManager, 203
- getPublishId
 - cbe::Publish, 200
 - cbe::Subscribe, 235
- getQuery
 - cbe::Filter, 147
 - cbe::GroupFilter, 159
- getQueryResult
 - cbe::QueryChain, 205
- getSecurity
 - cbe::Publish, 200
 - cbe::Subscribe, 236
- getShareFromUserId
 - cbe::Item, 168
- getShareIds
 - cbe::Item, 168
- getStreams
 - cbe::Object, 192
- GetStreamsDelegatePtr
 - cbe::delegate, 44
 - cbe::Object, 188
- getSubscribe
 - cbe::Item, 168
- getSubscribed
 - cbe::Subscribe, 236
- getSubscriptions
 - cbe::SubscribeManager, 238
- GetSubscriptionsDelegatePtr
 - cbe::delegate, 44
 - cbe::SubscribeManager, 237
- getTenantId
 - cbe::GroupManager, 161
- getTitle
 - cbe::Publish, 200
 - cbe::Subscribe, 236
- getUserFromShareId
 - cbe::Item, 168
- getVisibility
 - cbe::Group, 155
- Group
 - cbe::Group, 154
- groupContainer
 - cbe::Group, 155
- groupId
 - cbe::Member, 183
 - cbe::Role, 228
- GroupManager
 - cbe::GroupManager, 161
- groupManager
 - cbe::CloudBackend, 62
- GroupsLoaded
 - cbe::GroupQueryResult, 163
- hasPublished
 - cbe::Item, 169
- hasSubscribe
 - cbe::Item, 169
- ICloudBackendListenerPtr
 - cbe::delegate, 44
- id
 - cbe::Group, 155
 - cbe::Item, 169
 - cbe::Role, 228
- idLoaded
 - cbe::Item, 169
- itemsLoaded
 - cbe::QueryResult, 217
- ItemType
 - cbe, 38
- join
 - cbe::Group, 155
 - cbe::QueryChain, 206
 - cbe::QueryChainExt, 208–210
- JoinDelegatePtr
 - cbe::delegate, 44
 - cbe::delegate::group, 49
 - cbe::Group, 153
- joined
 - cbe::Group, 155
- KeyValues
 - cbe, 36
- kick
 - cbe::Member, 183
- KickDelegatePtr
 - cbe::delegate, 44
 - cbe::Member, 182
- leave
 - cbe::Group, 155
- LeaveDelegatePtr
 - cbe::delegate, 44
 - cbe::Group, 153
- length
 - cbe::Stream, 234
- libContainerId
 - cbe::Account, 52
- listAvailableShares
 - cbe::ShareManager, 232
- listBannedMembers
 - cbe::Group, 156
- ListBannedMembersDelegatePtr
 - cbe::Group, 153
- listGroups
 - cbe::GroupManager, 161
- ListGroupsDelegatePtr
 - cbe::delegate, 45
 - cbe::GroupManager, 161
- listMembers
 - cbe::Group, 156
 - cbe::Role, 229
- ListMembersDelegatePtr

- cbe::delegate, 45
- cbe::Group, 153
- cbe::Role, 228
- listMyShares
 - cbe::ShareManager, 233
- listRoles
 - cbe::Group, 156
- ListRolesDelegatePtr
 - cbe::delegate, 45
 - cbe::Group, 153
- ListSharesDelegatePtr
 - cbe::delegate, 45
 - cbe::ShareManager, 232
- login
 - cbe::CloudBackend, 63
- LoginDelegatePtr
 - cbe::CloudBackend, 60
 - cbe::delegate, 45
- Member
 - cbe::Member, 183
- MemberBanInfo
 - cbe, 37
- memberId
 - cbe::Member, 183
- message
 - cbe::delegate::Error, 96
- move
 - cbe::Container, 77
 - cbe::Object, 193
- MoveDelegatePtr
 - cbe::Container, 74
 - cbe::delegate::container, 48
 - cbe::delegate::object, 50
 - cbe::Object, 188
- name
 - cbe::Database, 89
 - cbe::Group, 156
 - cbe::Item, 169
 - cbe::Member, 183
 - cbe::Role, 229
- objectsLoaded
 - cbe::QueryResult, 217
- ObjectType
 - cbe, 38
- oldParentId
 - cbe::Item, 169
- onAclError
 - cbe::delegate::AclDelegate, 54
- onAclSuccess
 - cbe::delegate::AclDelegate, 54
- onAddRoleMemberError
 - cbe::delegate::AddRoleMemberDelegate, 55
- onAddRoleMemberSuccess
 - cbe::delegate::AddRoleMemberDelegate, 55
- onBanError
 - cbe::delegate::BanDelegate, 56
- onBanSuccess
 - cbe::delegate::BanDelegate, 57
- onChunkReceived
 - cbe::delegate::DownloadBinaryDelegate, 90
 - cbe::delegate::DownloadDelegate, 92
- onChunkSent
 - cbe::delegate::UploadDelegate, 248
- onCreateAccountError
 - cbe::delegate::CreateAccountDelegate, 84
- onCreateAccountSuccess
 - cbe::delegate::CreateAccountDelegate, 84
- onCreateContainerError
 - cbe::delegate::CreateContainerDelegate, 85
- onCreateContainerSuccess
 - cbe::delegate::CreateContainerDelegate, 85
- onCreateGroupError
 - cbe::delegate::CreateGroupDelegate, 86
- onCreateGroupSuccess
 - cbe::delegate::CreateGroupDelegate, 86
- onCreateObjectError
 - cbe::delegate::CreateObjectDelegate, 86
- onCreateObjectSuccess
 - cbe::delegate::CreateObjectDelegate, 86
- onCreateRoleError
 - cbe::delegate::CreateRoleDelegate, 87
- onCreateRoleSuccess
 - cbe::delegate::CreateRoleDelegate, 87
- onDownloadBinaryError
 - cbe::delegate::DownloadBinaryDelegate, 90
- onDownloadBinarySuccess
 - cbe::delegate::DownloadBinaryDelegate, 91
- onDownloadError
 - cbe::delegate::DownloadDelegate, 93
- onDownloadSuccess
 - cbe::delegate::DownloadDelegate, 93
- onGetStreamsError
 - cbe::delegate::GetStreamsDelegate, 150
- onGetStreamsSuccess
 - cbe::delegate::GetStreamsDelegate, 150
- onGetSubscriptionsError
 - cbe::delegate::GetSubscriptionsDelegate, 151
- onGetSubscriptionsSuccess
 - cbe::delegate::GetSubscriptionsDelegate, 151
- onItemError
 - cbe::delegate::ItemError, 170
- onJoinError
 - cbe::delegate::group::JoinDelegate, 171
 - cbe::delegate::JoinDelegate, 172
 - cbe::delegate::JoinError, 173
- onJoinSuccess
 - cbe::delegate::group::JoinDelegate, 171
 - cbe::delegate::JoinDelegate, 172
- onKickError
 - cbe::delegate::KickDelegate, 174
- onKickSuccess
 - cbe::delegate::KickDelegate, 174
- onLeaveError
 - cbe::delegate::LeaveDelegate, 175

- onLeaveSuccess
 - cbe::delegate::LeaveDelegate, 175
- onListGroupError
 - cbe::delegate::ListGroupDelegate, 176
- onListGroupSuccess
 - cbe::delegate::ListGroupDelegate, 176
- onListMembersError
 - cbe::delegate::ListMembersDelegate, 177
- onListMembersSuccess
 - cbe::delegate::ListMembersDelegate, 177
- onListRolesError
 - cbe::delegate::ListRolesDelegate, 178
- onListRolesSuccess
 - cbe::delegate::ListRolesDelegate, 178
- onListSharesError
 - cbe::delegate::ListSharesDelegate, 179
- onListSharesSuccess
 - cbe::delegate::ListSharesDelegate, 179
- onLoadError
 - cbe::delegate::LoadError, 179
- onLoginError
 - cbe::delegate::LoginDelegate, 180
- onLoginSuccess
 - cbe::delegate::LoginDelegate, 181
- onMoveError
 - cbe::delegate::container::MoveDelegate, 185
 - cbe::delegate::object::MoveDelegate, 185
- onMoveSuccess
 - cbe::delegate::container::MoveDelegate, 185
 - cbe::delegate::object::MoveDelegate, 185
- onPublishError
 - cbe::delegate::PublishDelegate, 202
- onPublishSuccess
 - cbe::delegate::PublishDelegate, 202
- onQueryError
 - cbe::delegate::QueryDelegate, 212
- onQueryJoinError
 - cbe::delegate::QueryJoinDelegate, 214
- onQueryJoinSuccess
 - cbe::delegate::QueryJoinDelegate, 214
- onQuerySuccess
 - cbe::delegate::QueryDelegate, 212
 - cbe::delegate::QueryLoaded, 215
- onRemoteContainerAdded
 - cbe::delegate::CloudBackendListenerDelegate, 70
 - cbe::delegate::ICloudBackendListener, 164
- onRemoteContainerMoved
 - cbe::delegate::CloudBackendListenerDelegate, 70
 - cbe::delegate::ICloudBackendListener, 164
- onRemoteContainerRemoved
 - cbe::delegate::CloudBackendListenerDelegate, 70
 - cbe::delegate::ICloudBackendListener, 165
- onRemoteContainerRenamed
 - cbe::delegate::CloudBackendListenerDelegate, 70
 - cbe::delegate::ICloudBackendListener, 165
- onRemoteObjectAdded
 - cbe::delegate::CloudBackendListenerDelegate, 71
 - cbe::delegate::ICloudBackendListener, 165
- onRemoteObjectMoved
 - cbe::delegate::CloudBackendListenerDelegate, 71
 - cbe::delegate::ICloudBackendListener, 165
- onRemoteObjectRemoved
 - cbe::delegate::CloudBackendListenerDelegate, 71
 - cbe::delegate::ICloudBackendListener, 165
- onRemoteObjectRenamed
 - cbe::delegate::CloudBackendListenerDelegate, 71
 - cbe::delegate::ICloudBackendListener, 166
- onRemoveError
 - cbe::delegate::container::RemoveDelegate, 218
 - cbe::delegate::group::RemoveDelegate, 219
 - cbe::delegate::object::RemoveDelegate, 219
- onRemoveRoleError
 - cbe::delegate::RemoveRoleDelegate, 220
- onRemoveRoleMemberError
 - cbe::delegate::RemoveRoleMemberDelegate, 221
- onRemoveRoleMemberSuccess
 - cbe::delegate::RemoveRoleMemberDelegate, 221
- onRemoveRoleSuccess
 - cbe::delegate::RemoveRoleDelegate, 220
- onRemoveSuccess
 - cbe::delegate::container::RemoveDelegate, 218
 - cbe::delegate::group::RemoveDelegate, 219
 - cbe::delegate::object::RemoveDelegate, 220
- onRenameError
 - cbe::delegate::container::RenameDelegate, 223
 - cbe::delegate::group::RenameDelegate, 224
 - cbe::delegate::object::RenameDelegate, 225
- onRenameSuccess
 - cbe::delegate::container::RenameDelegate, 223
 - cbe::delegate::group::RenameDelegate, 224
 - cbe::delegate::object::RenameDelegate, 225
- onSearchGroupsError
 - cbe::delegate::SearchGroupsDelegate, 230
- onSearchGroupsSuccess
 - cbe::delegate::SearchGroupsDelegate, 230
- onShareError
 - cbe::delegate::ShareDelegate, 231
- onShareSuccess
 - cbe::delegate::ShareDelegate, 231
- onSubscribeError
 - cbe::delegate::SubscribeDelegate, 236
- onSubscribeSuccess
 - cbe::delegate::SubscribeDelegate, 237
- onUnBanError
 - cbe::delegate::UnBanDelegate, 241
- onUnBanSuccess
 - cbe::delegate::UnBanDelegate, 241
- onUnPublishError
 - cbe::delegate::UnPublishDelegate, 242
- onUnPublishSuccess
 - cbe::delegate::UnPublishDelegate, 242
- onUnShareError
 - cbe::delegate::UnShareDelegate, 244
- onUnShareSuccess
 - cbe::delegate::UnShareDelegate, 244
- onUnSubscribeError

- cbe::delegate::UnSubscribeDelegate, 245
- onUnSubscribeSuccess
 - cbe::delegate::UnSubscribeDelegate, 245
- onUpdateKeyValuesError
 - cbe::delegate::UpdateKeyValuesDelegate, 247
- onUpdateKeyValuesSuccess
 - cbe::delegate::UpdateKeyValuesDelegate, 247
- onUploadError
 - cbe::delegate::UploadDelegate, 248
- onUploadSuccess
 - cbe::delegate::UploadDelegate, 248
- operator bool
 - cbe::Account, 52
 - cbe::CloudBackend, 64
 - cbe::delegate::DownloadBinarySuccess, 92
 - cbe::delegate::DownloadSuccess, 94
 - cbe::delegate::Error, 95
 - cbe::delegate::group::RenameSuccess, 226
 - cbe::delegate::LeaveSuccess, 175
 - cbe::delegate::object::RemoveSuccess, 223
 - cbe::delegate::PublishSuccess, 204
 - cbe::delegate::ShareSuccess, 234
 - cbe::delegate::UnPublishSuccess, 243
 - cbe::delegate::UnShareSuccess, 244
 - cbe::delegate::UnSubscribeSuccess, 246
 - cbe::Group, 156
 - cbe::GroupManager, 161
 - cbe::Item, 169
 - cbe::Member, 184
 - cbe::Publish, 200
 - cbe::PublishManager, 203
 - cbe::QueryResult, 217
 - cbe::Role, 229
 - cbe::ShareManager, 233
 - cbe::util::Optional< T >, 198
- operator<<
 - cbe::delegate::Error, 95
 - cbe::GroupFilter, 160
- ownerId
 - cbe::Database, 89
 - cbe::Item, 169
- parentId
 - cbe::Group, 157
 - cbe::Item, 169
- password
 - cbe::Account, 52
- path
 - cbe::Item, 169
- Permissions
 - cbe, 38
- ProgressEventFn
 - cbe::delegate, 45
- Publish
 - cbe::Publish, 200
- publish
 - cbe::Container, 77
 - cbe::Object, 193
- PublishAccess
 - cbe, 39
- PublishDelegatePtr
 - cbe::Container, 74
 - cbe::delegate, 45
 - cbe::Object, 189
- PublishManager
 - cbe::PublishManager, 203
- publishManager
 - cbe::CloudBackend, 64
- PublishVisibility
 - cbe, 39
- query
 - cbe::CloudBackend, 65, 66
 - cbe::Container, 77, 78
- QueryDelegatePtr
 - cbe::CloudBackend, 60
 - cbe::Container, 74
 - cbe::delegate, 45
- QueryJoinDelegatePtr
 - cbe::CloudBackend, 60
 - cbe::Container, 75
 - cbe::delegate, 46
- QueryResult
 - cbe::QueryResult, 216
- queryWithPath
 - cbe::CloudBackend, 66, 67
 - cbe::Container, 78
- readLocked
 - cbe::Database, 89
- reason
 - cbe::delegate::Error, 96
- remove
 - cbe::Container, 78
 - cbe::Group, 157
 - cbe::Object, 193
- RemoveDelegatePtr
 - cbe::Container, 75
 - cbe::delegate::container, 48
 - cbe::delegate::group, 49
 - cbe::delegate::object, 50
 - cbe::Group, 153
 - cbe::Object, 189
- removeListener
 - cbe::CloudBackend, 67
- removeRole
 - cbe::Group, 157
- RemoveRoleDelegatePtr
 - cbe::delegate, 46
 - cbe::Group, 153
- removeRoleMember
 - cbe::Role, 229
- RemoveRoleMemberDelegatePtr
 - cbe::delegate, 46
 - cbe::Role, 228
- rename
 - cbe::Container, 79
 - cbe::Group, 157

- cbe::Object, 194
- RenameDelegatePtr
 - cbe::Container, 75
 - cbe::delegate::container, 48
 - cbe::delegate::group, 49
 - cbe::delegate::object, 50
 - cbe::Group, 153
 - cbe::Object, 189
- requests
 - cbe::Group, 157
- Role
 - cbe::Role, 228
- rootContainer
 - cbe::Account, 53
 - cbe::Database, 89
- rootDatabase
 - cbe::Account, 53
- search
 - cbe::CloudBackend, 67, 68
 - cbe::Container, 79
- searchGroups
 - cbe::GroupManager, 162
- SearchGroupsDelegatePtr
 - cbe::delegate, 46
 - cbe::GroupManager, 161
- setAcl
 - cbe::Container, 80
 - cbe::Object, 194
- SetAclDelegatePtr
 - cbe::Container, 75
 - cbe::Object, 189
- setAscending
 - cbe::Filter, 147
 - cbe::GroupFilter, 159
- setByPassCache
 - cbe::Filter, 148
- setContainerFirst
 - cbe::Filter, 148
- setCount
 - cbe::Filter, 148
 - cbe::GroupFilter, 159
- setDataType
 - cbe::Filter, 148
- setDeleted
 - cbe::Filter, 148
 - cbe::GroupFilter, 159
- setDescription
 - cbe::Publish, 201
- setFilter
 - cbe::GroupFilter, 159
- setItemOrder
 - cbe::Filter, 149
- setOffset
 - cbe::Filter, 149
 - cbe::GroupFilter, 159
- setPassword
 - cbe::Publish, 201
- setPrivacy
 - cbe::Publish, 201
- setQuery
 - cbe::Filter, 149
 - cbe::GroupFilter, 160
- setSecurity
 - cbe::Publish, 201
- setTitle
 - cbe::Publish, 201
- share
 - cbe::Container, 80
 - cbe::Object, 194
- ShareDelegatePtr
 - cbe::Container, 75
 - cbe::delegate, 46
 - cbe::Object, 189
- ShareManager
 - cbe::ShareManager, 232
- shareManager
 - cbe::CloudBackend, 68
- source
 - cbe::Account, 53
- streamId
 - cbe::Stream, 234
- Streams
 - cbe::Object, 189
- subscribe
 - cbe::SubscribeManager, 238, 239
- SubscribeDelegatePtr
 - cbe::delegate, 46
 - cbe::SubscribeManager, 238
- subscribeManager
 - cbe::CloudBackend, 68
- Success
 - cbe::delegate::LoginDelegate, 180
- tenantContainerId
 - cbe::Account, 53
- terminate
 - cbe::CloudBackend, 68
- totalCount
 - cbe::GroupQueryResult, 163
 - cbe::QueryResult, 217
- type
 - cbe::Item, 170
- unBan
 - cbe::Member, 184
- UnBanDelegatePtr
 - cbe::delegate, 46
 - cbe::Member, 182
- unPublish
 - cbe::Container, 80
 - cbe::Object, 195
- UnPublishDelegatePtr
 - cbe::Container, 75
 - cbe::delegate, 47
 - cbe::Object, 189
- unShare
 - cbe::Container, 81

- cbe::Object, [195](#)
- UnShareDelegatePtr
 - cbe::Container, [75](#)
 - cbe::delegate, [47](#)
 - cbe::Object, [189](#)
- unSubscribe
 - cbe::Container, [81](#)
 - cbe::Object, [195](#)
 - cbe::Subscribe, [236](#)
- UnSubscribeDelegatePtr
 - cbe::Container, [75](#)
 - cbe::delegate, [47](#)
 - cbe::Object, [189](#)
- updated
 - cbe::Item, [170](#)
- updateKeyValues
 - cbe::Object, [195](#), [196](#)
- UpdateKeyValuesDelegatePtr
 - cbe::delegate, [47](#)
 - cbe::Object, [189](#)
- upload
 - cbe::Container, [81–83](#)
- UploadDelegatePtr
 - cbe::Container, [75](#)
 - cbe::delegate, [47](#)
 - cbe::Object, [189](#)
- uploadStream
 - cbe::Object, [196](#)
- username
 - cbe::Account, [53](#)
 - cbe::Item, [170](#)
- version
 - cbe::CloudBackend, [69](#)
- Visibility
 - cbe, [40](#)
- visibility
 - cbe::Member, [184](#)
- writeLocked
 - cbe::Database, [89](#)