Rest Demo App

Generated by Doxygen 1.8.13

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

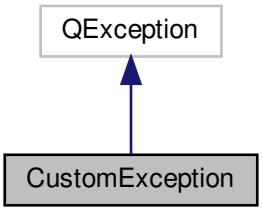
Class Documentation

4.1 CustomException Class Reference

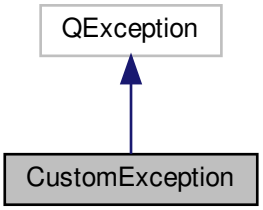
The CustomException class implements a custom Exception used to handle runtime errors inside the application.

`#include <customexception.h>`

Inheritance diagram for CustomException:



Collaboration diagram for CustomException:



Public Member Functions

- CustomException (const QString &message)

  Class constructor.
- virtual ~CustomException ()

  Class destructor.
- QString getMessage () const

4.1.1 Detailed Description

The CustomException class implements a custom Exception used to handle runtime errors inside the application.

Definition at line 17 of file customexception.h.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 CustomException()

```
CustomException::CustomException (
            const QString & message )  [inline], [explicit]
```

Class constructor.

Parameters

message	The exception message.

Definition at line 25 of file customexception.h.

4.1.3 Member Function Documentation

4.1.3.1 getMessage()

```
QString CustomException::getMessage ( ) const  [inline]
```

Returns

the exception message.

Definition at line 39 of file customexception.h.

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

- customexception.h

4.2 DBController Class Reference

The DBController class implements the API to handle an SQLite database using the native Qt API for SQL databases. This controller only provides methods to work with content data as specified on the requeriments. This class des not implements any business logic.

```
#include <dbcontroller.h>
```

Public Member Functions

- DBController (QSqlDatabase *database)
  Class constructor.
- void close ()
  Closes the database if it is open.
- bool registerContent (const Content &content, QString *errorMessage)
  Registers (creates) a new Content entrance on the database.
- bool deleteContent (const qint64 &id, QString *errorMessage)
  Deletes a Content entrance from the database.
- bool updateContent (const Content &content, QString *errorMessage)
  Updates a Content entrance on the database.
- bool getContent (const qint64 &id, Content *content, QString *errorMessage)
  Retrieves Content data from the database.
- bool getDevice (const qint64 &id, Device *device, QString *errorMessage)
  Retrieves Device data from the database.
- bool getProtectionSystem (const qint64 &id, ProtectionSystem *protectionSystem, QString *errorMessage)
  Retrieves Protection System data from the database.

Static Public Attributes

- static const QString DB_CONNECTION_NAME = "QSQLITE"
  The kind of connection to the database.

4.2.1 Detailed Description

The DBController class implements the API to handle an SQLite database using the native Qt API for SQL databases. This controller only provides methods to work with content data as specified on the requeriments. This class des not implements any business logic.

Definition at line 24 of file dbcontroller.h.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 DBController()

```
DBController::DBController (
            QSqlDatabase * database )
```

Class constructor.

Parameters

database	The database instance to use.

Definition at line 21 of file dbcontroller.cpp.

4.2.3 Member Function Documentation

4.2.3.1 deleteContent()

```
bool DBController::deleteContent (
            const qint64 & id,
            QString * errorMessage )
```

Deletes a Content entrance from the database.

Parameters

id	The Content id of the Content data to remove from the database.
errorMessage	The resulting error message.

Returns

True if no error occurred, false otherwise.

Definition at line 82 of file dbcontroller.cpp.

4.2.3.2 getContent()

```
bool DBController::getContent (
            const qint64 & id,
            Content * content,
            QString * errorMessage )
```

Retrieves Content data from the database.

Parameters

id	The Content id of the Content data to get from the database.
errorMessage	The resulting error message.

Returns

True if no error occurred, false otherwise.

Definition at line 141 of file dbcontroller.cpp.

4.2.3.3 getDevice()

```
bool DBController::getDevice (
            const qint64 & id,
            Device * device,
            QString * errorMessage )
```

Retrieves Device data from the database.

Parameters

id	The Device id of the Content data to get from the database.
errorMessage	The resulting error message.

Returns

True if no error occurred, false otherwise.

Definition at line 174 of file dbcontroller.cpp.

4.2.3.4 getProtectionSystem()

```
bool DBController::getProtectionSystem (
            const qint64 & id,
            ProtectionSystem * protectionSystem,
            QString * errorMessage )
```

Retrieves Protection System data from the database.

Parameters

id	The Protection System id of the Content data to get from the database.
errorMessage	The resulting error message.

Returns

True if no error occurred, false otherwise.

Definition at line 204 of file dbcontroller.cpp.

4.2.3.5 registerContent()

```
bool DBController::registerContent (
            const Content & content,
            QString * errorMessage )
```

Registers (creates) a new Content entrance on the database.

Parameters

content	The Content data to insert on the database.
errorMessage	The resulting error message.

Returns

True if no error occurred, false otherwise.

Definition at line 50 of file dbcontroller.cpp.

4.2.3.6 updateContent()

```
bool DBController::updateContent (
            const Content & content,
            QString * errorMessage )
```

Updates a Content entrance on the database.

Parameters

content	The new Content data to write on the database.
errorMessage	The resulting error message.

Returns

True if no error occurred, false otherwise.

Definition at line 99 of file dbcontroller.cpp.

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

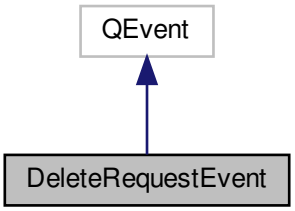
- dbcontroller.h
- dbcontroller.cpp

4.3 DeleteRequestEvent Class Reference

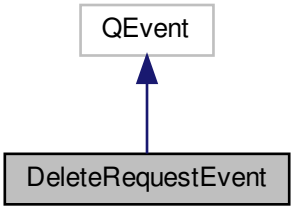
The DeleteRequestEvent class declares the message to notify a new Delete request.

`#include <requestevents.h>`

Inheritance diagram for DeleteRequestEvent:



Collaboration diagram for DeleteRequestEvent:



Public Member Functions

- DeleteRequestEvent (const quint64 &requestId, const qint64 &id)
  Class constructor.
- quint64 getRequestId () const noexcept
- qint64 getContentId () const
- QString toString () const

Static Public Attributes

- static const QEvent::Type ID
  The message ID.

4.3.1 Detailed Description

The DeleteRequestEvent class declares the message to notify a new Delete request.

Definition at line 121 of file requestevents.h.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 DeleteRequestEvent()

```
DeleteRequestEvent::DeleteRequestEvent (
            const quint64 & requestId,
            const qint64 & id )  [inline], [explicit]
```

Class constructor.

Parameters

requestId	The request identifier.
id	The content id to remove.

Definition at line 131 of file requestevents.h.

4.3.3 Member Function Documentation

4.3.3.1 getContentId()

```
qint64 DeleteRequestEvent::getContentId ( ) const  [inline]
```

Returns

the content id to remove.

Definition at line 146 of file requestevents.h.

4.3.3.2 getRequestId()

```
quint64 DeleteRequestEvent::getRequestId ( ) const  [inline], [noexcept]
```

Returns

the request id this message belongs to.

Definition at line 139 of file requestevents.h.

4.3.3.3 toString()

```
QString DeleteRequestEvent::toString ( ) const  [inline]
```

Returns

message description.

Definition at line 153 of file requestevents.h.

4.3.4 Member Data Documentation

4.3.4.1 ID

```
const QEvent::Type DeleteRequestEvent::ID  [static]
```

Initial value:

```
=
    static_cast<QEvent::Type>(QEvent::registerEventType())
```

The message ID.

Definition at line 124 of file requestevents.h.

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

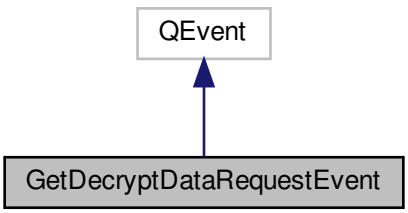
- requestevents.h
- requestevents.cpp

4.4 GetDecryptDataRequestEvent Class Reference

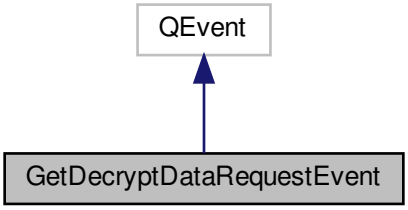
The GetDecryptDataRequestEvent class declares the message to notify a new request to get some content data decrypted.

```
#include <requestevents.h>
```

Inheritance diagram for GetDecryptDataRequestEvent:



Collaboration diagram for GetDecryptDataRequestEvent:



Public Member Functions

- GetDecryptDataRequestEvent (const quint64 &requestId, const qint64 &deviceId, const qint64 &contentId)
  Class constructor.
- quint64 getRequestId () const noexcept
- qint64 getDeviceId () const
- qint64 getContentId () const
- QString toString () const

Static Public Attributes

- static const QEvent::Type ID
  The message ID.

4.4.1 Detailed Description

The GetDecryptDataRequestEvent class declares the message to notify a new request to get some content data decrypted.

Definition at line 211 of file requestevents.h.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 GetDecryptDataRequestEvent()

```
GetDecryptDataRequestEvent::GetDecryptDataRequestEvent (
            const quint64 & requestId,
            const qint64 & deviceId,
            const qint64 & contentId )  [inline], [explicit]
```

Class constructor.

Parameters

deviceId	The device id.
contentId	The content id to decrypt.

Definition at line 221 of file requestevents.h.

4.4.3 Member Function Documentation

4.4.3.1 getContentId()

```
qint64 GetDecryptDataRequestEvent::getContentId ( ) const  [inline]
```

Returns

the content id to decrypt.

Definition at line 243 of file requestevents.h.

4.4.3.2 getDeviceId()

```
qint64 GetDecryptDataRequestEvent::getDeviceId ( ) const  [inline]
```

Returns

the device id.

Definition at line 236 of file requestevents.h.

4.4.3.3 getRequestId()

```
quint64 GetDecryptDataRequestEvent::getRequestId ( ) const  [inline], [noexcept]
```

Returns

the request id this message belongs to.

Definition at line 229 of file requestevents.h.

4.4.3.4 toString()

```
QString GetDecryptDataRequestEvent::toString ( ) const  [inline]
```

Returns

message description.

Definition at line 250 of file requestevents.h.

4.4.4 Member Data Documentation

4.4.4.1 ID

```
const QEvent::Type GetDecryptDataRequestEvent::ID  [static]
```

Initial value:

```
=
    static_cast<QEvent::Type>(QEvent::registerEventType())
```

The message ID.

Definition at line 214 of file requestevents.h.

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

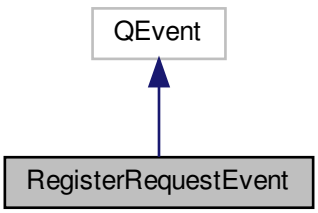
- requestevents.h
- requestevents.cpp

4.5 RegisterRequestEvent Class Reference

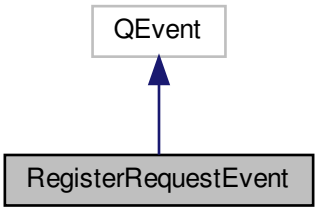
The RegisterRequestEvent class declares the message to notify a new Register request.

```
#include <requestevents.h>
```

Inheritance diagram for RegisterRequestEvent:



Collaboration diagram for RegisterRequestEvent:



Public Member Functions

- RegisterRequestEvent (const quint64 &requestId, const Content &content)
  Class constructor.
- quint64 getRequestId () const noexcept
- Content getContent () const
- QString toString () const

Static Public Attributes

- static const QEvent::Type ID
  The message ID.

4.5.1 Detailed Description

The RegisterRequestEvent class declares the message to notify a new Register request.

Definition at line 31 of file requestevents.h.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 RegisterRequestEvent()

```
RegisterRequestEvent::RegisterRequestEvent (
            const quint64 & requestId,
            const Content & content )  [inline], [explicit]
```

Class constructor.

Parameters

requestId	The request identifier.
content	The content to insert on the database.

Definition at line 41 of file requestevents.h.

4.5.3 Member Function Documentation

4.5.3.1 getContent()

```
Content RegisterRequestEvent::getContent ( ) const  [inline]
```

Returns

the content to insert on the database.

Definition at line 56 of file requestevents.h.

4.5.3.2 getRequestId()

```
quint64 RegisterRequestEvent::getRequestId ( ) const  [inline], [noexcept]
```

Returns

the request id this message belongs to.

Definition at line 49 of file requestevents.h.

4.5.3.3 toString()

```
QString RegisterRequestEvent::toString ( ) const  [inline]
```

Returns

message description.

Definition at line 63 of file requestevents.h.

4.5.4 Member Data Documentation

4.5.4.1 ID

```
const QEvent::Type RegisterRequestEvent::ID  [static]
```

Initial value:

```
=
    static_cast<QEvent::Type>(QEvent::registerEventType())
```

The message ID.

Definition at line 34 of file requestevents.h.

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

- requestevents.h
- requestevents.cpp

4.6 RequestsManager Class Reference

The RequestsManager class implements the business logic of the application. It recieves the requests notified by the endpoints and uses the Database Controller to handle the persistent data.

```
#include <requestsmanager.h>
```

Public Member Functions

- RequestsManager (EndpointRegister *ep_register, EndpointDelete *ep_delete, EndpointUpdate *ep_update, EndpointView *ep_view, EndpointGetDecrypData *ep_get_decryp_data, DBController *db_controller)

  Class contructor.
- ~RequestsManager ()

  Class destructor.
- void onRegisterRequestEvent (const quint64 &requestId, const Content &content)

  Handles the RegisterRequestEvent message. This method tells the Database Controller to write the data and then reponds the request through the proper endpoint.
- void onDeleteRequestEvent (const quint64 &requestId, const qint64 &id)

  Handles the DeleteRequestEvent message. This method tells the Database Controller to remove the data and then reponds the request through the proper endpoint.
- void onUpdateRequestEvent (const quint64 &requestId, const Content &content)

  Handles the UpdateRequestEvent message. This method tells the Database Controller to update the data and then reponds the request through the proper endpoint.
- void onViewRequestEvent (const quint64 &requestId, const qint64 &id)

  Handles the ViewRequestEvent message. This method tells the Database Controller to get the data and then reponds the request through the proper endpoint.
- void onGetDecryptDataRequestEvent (const quint64 &requestId, const qint64 &deviceId, const qint64 &contentId)

  Handles the GetDecryptDataRequestEvent message. This method checks if the Content Protection System is the same as the Device protection system. If it is then decrypts the Content payload and responds the request with this data.

4.6.1 Detailed Description

The RequestsManager class implements the business logic of the application. It recieves the requests notified by the endpoints and uses the Database Controller to handle the persistent data.

Definition at line 26 of file requestsmanager.h.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 RequestsManager()

```
RequestsManager::RequestsManager (
            EndpointRegister * ep_register,
            EndpointDelete * ep_delete,
            EndpointUpdate * ep_update,
            EndpointView * ep_view,
            EndpointGetDecrypData * ep_get_decryp_data,
            DBController * db_controller )  [explicit]
```

Class contructor.

Parameters

ep_register	The endpoint for register requests to use.
ep_delete	The endpoint for delete requests to use.
ep_update	The endpoint for update requests to use.
ep_view	The endpoint for view requests to use.
ep_get_decryp_data	The endpoint for decrypted content requests to use.
db_controller	The Databse Controller to use.

Definition at line 11 of file requestsmanager.cpp.

4.6.3 Member Function Documentation

4.6.3.1 onDeleteRequestEvent()

```
void RequestsManager::onDeleteRequestEvent (
            const quint64 & requestId,
            const qint64 & id )
```

Handles the DeleteRequestEvent message. This method tells the Database Controller to remove the data and then reponds the request through the proper endpoint.

Parameters

requestId	The request identifier to respond to.
id	The Content identifier to remove from the Database.

Definition at line 76 of file requestsmanager.cpp.

4.6.3.2 onGetDecryptDataRequestEvent()

```
void RequestsManager::onGetDecryptDataRequestEvent (
            const quint64 & requestId,
            const qint64 & deviceId,
            const qint64 & contentId )
```

Handles the GetDecryptDataRequestEvent message. This method checks if the Content Protection System is the same as the Device protection system. If it is then decrypts the Content payload and responds the request with this data.

Parameters

requestId	The request identifier to respond to.
deviceId	The device identifier the content msut be played to.
contentId	The Content identifier to decrypt.

Definition at line 101 of file requestsmanager.cpp.

4.6.3.3 onRegisterRequestEvent()

```
void RequestsManager::onRegisterRequestEvent (
            const quint64 & requestId,
            const Content & content )
```

Handles the RegisterRequestEvent message. This method tells the Database Controller to write the data and then reponds the request through the proper endpoint.

Parameters

requestId	The request identifier to respond to.
content	The Content to write on the Database.

Definition at line 55 of file requestsmanager.cpp.

4.6.3.4 onUpdateRequestEvent()

```
void RequestsManager::onUpdateRequestEvent (
            const quint64 & requestId,
            const Content & content )
```

Handles the UpdateRequestEvent message. This method tells the Database Controller to update the data and then reponds the request through the proper endpoint.

Parameters

requestId	The request identifier to respond to.
content	The new Content to write on the Database.

Definition at line 84 of file requestsmanager.cpp.

4.6.3.5 onViewRequestEvent()

```
void RequestsManager::onViewRequestEvent (
            const quint64 & requestId,
            const qint64 & id )
```

Handles the ViewRequestEvent message. This method tells the Database Controller to get the data and then reponds the request through the proper endpoint.

Parameters

requestId	The request identifier to respond to.
content	The Content to write on the Database

Definition at line 92 of file requestsmanager.cpp.

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

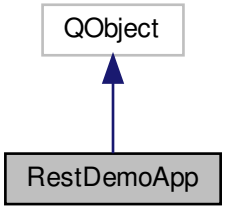
- requestsmanager.h
- requestsmanager.cpp

4.7 RestDemoApp Class Reference

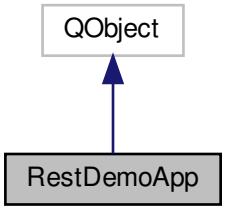
The RestDemoApp class implements the initialization of the application. In addition, this class receives the messages from the endpoints notifying new requests and delegates them to the Requests Manager. So, this class does not implements any business logic.

```
#include <restdemoapp.h>
```

Inheritance diagram for RestDemoApp:



Collaboration diagram for RestDemoApp:



Public Slots

- bool event (QEvent *event)

  The method who handles all the QEvents received by this class.

Public Member Functions

- RestDemoApp ()

  Class constructor.
- ~RestDemoApp ()

  Class destructor.

4.7.1 Detailed Description

The RestDemoApp class implements the initialization of the application. In addition, this class receives the messages from the endpoints notifying new requests and delegates them to the Requests Manager. So, this class does not implements any business logic.

Definition at line 20 of file restdemoapp.h.

4.7.2 Member Function Documentation

4.7.2.1 event

```
bool RestDemoApp::event (
            QEvent * event )  [slot]
```

The method who handles all the QEvents received by this class.

Parameters

event	The event received.

Returns

True if the event must be handled by Qt, false otherwise.

Definition at line 40 of file restdemoapp.cpp.

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

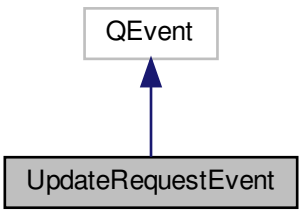
- restdemoapp.h
- restdemoapp.cpp

4.8 UpdateRequestEvent Class Reference

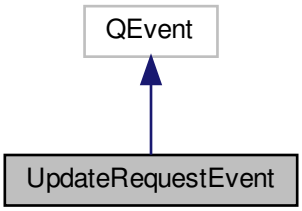
The UpdateRequestEvent class declares the message to notify a new Update request.

```
#include <requestevents.h>
```

Inheritance diagram for UpdateRequestEvent:



Collaboration diagram for UpdateRequestEvent:



Public Member Functions

- UpdateRequestEvent (const quint64 &requestId, const Content &content)
  Class constructor.
- quint64 getRequestId () const noexcept
- Content getContent () const
- QString toString () const

Static Public Attributes

- static const QEvent::Type ID
  The message ID.

4.8.1 Detailed Description

The UpdateRequestEvent class declares the message to notify a new Update request.

Definition at line 76 of file requestevents.h.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 UpdateRequestEvent()

```
UpdateRequestEvent::UpdateRequestEvent (
            const quint64 & requestId,
            const Content & content )  [inline], [explicit]
```

Class constructor.

Parameters

requestId	The request identifier.
content	The content to insert on the database.

Definition at line 86 of file requestevents.h.

4.8.3 Member Function Documentation

4.8.3.1 getContent()

```
Content UpdateRequestEvent::getContent ( ) const  [inline]
```

Returns

the new content to write on the database.

Definition at line 101 of file requestevents.h.

4.8.3.2 getRequestId()

```
quint64 UpdateRequestEvent::getRequestId ( ) const  [inline], [noexcept]
```

Returns

the request id this message belongs to.

Definition at line 94 of file requestevents.h.

4.8.3.3 toString()

```
QString UpdateRequestEvent::toString ( ) const  [inline]
```

Returns

message description.

Definition at line 108 of file requestevents.h.

4.8.4 Member Data Documentation

4.8.4.1 ID

```
const QEvent::Type UpdateRequestEvent::ID  [static]
```

Initial value:

```
=
    static_cast<QEvent::Type>(QEvent::registerEventType())
```

The message ID.

Definition at line 79 of file requestevents.h.

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

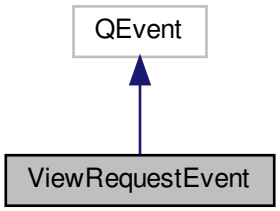
- requestevents.h
- requestevents.cpp

4.9 ViewRequestEvent Class Reference

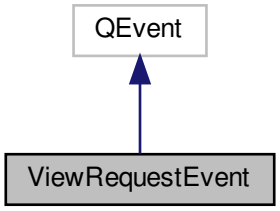
The ViewRequestEvent class declares the message to notify a new View request.

`#include <requestevents.h>`

Inheritance diagram for ViewRequestEvent:



Collaboration diagram for ViewRequestEvent:



Public Member Functions

- ViewRequestEvent (const quint64 &requestId, const qint64 &id)

  Class constructor.
- quint64 getRequestId () const noexcept
- qint64 getContentId () const
- QString toString () const

Static Public Attributes

- static const QEvent::Type ID

  The message ID.

4.9.1 Detailed Description

The ViewRequestEvent class declares the message to notify a new View request.

Definition at line 166 of file requestevents.h.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 ViewRequestEvent()

```
ViewRequestEvent::ViewRequestEvent (
            const quint64 & requestId,
            const qint64 & id )  [inline], [explicit]
```

Class constructor.

Parameters

requestId	The request identifier.
id	The content id to view.

Definition at line 176 of file requestevents.h.

4.9.3 Member Function Documentation

4.9.3.1 getContentId()

```
qint64 ViewRequestEvent::getContentId ( ) const  [inline]
```

Returns

the content id to view.

Definition at line 191 of file requestevents.h.

4.9.3.2 getRequestId()

```
quint64 ViewRequestEvent::getRequestId ( ) const  [inline], [noexcept]
```

Returns

the request id this message belongs to.

Definition at line 184 of file requestevents.h.

4.9.3.3 toString()

```
QString ViewRequestEvent::toString ( ) const  [inline]
```

Returns

message description.

Definition at line 198 of file requestevents.h.

4.9.4 Member Data Documentation

4.9.4.1 ID

```
const QEvent::Type ViewRequestEvent::ID  [static]
```

Initial value:

```
=
    static_cast<QEvent::Type>(QEvent::registerEventType())
```

The message ID.

Definition at line 169 of file requestevents.h.

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

- requestevents.h
- requestevents.cpp

Chapter 5

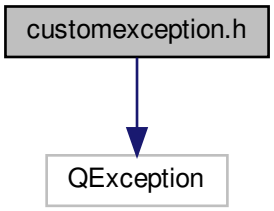
File Documentation

5.1 customexception.h File Reference

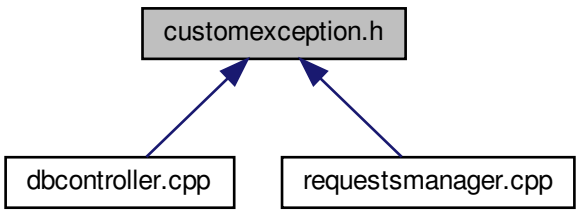
Custom Exception class declaration.

```
#include <QException>
```

Include dependency graph for customexception.h:



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



Classes

- class CustomException

  The CustomException class implements a custom Exception used to handle runtime errors inside the application.

5.1.1 Detailed Description

Custom Exception class declaration.

Author

Rubén Sánchez Castellano

Date

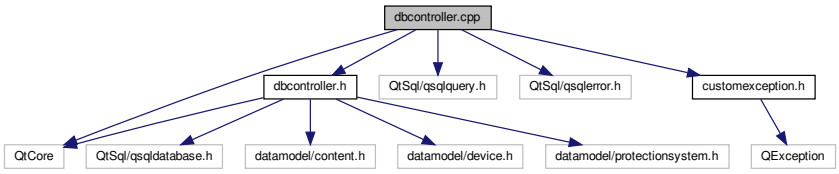
August 24, 2018

5.2 dbcontroller.cpp File Reference

DBController class definition.

```
#include "dbcontroller.h"
#include <QtSql/qsqlquery.h>
#include <QtSql/qsqlerror.h>
#include "customexception.h"
```

Include dependency graph for dbcontroller.cpp:



5.2.1 Detailed Description

DBController class definition.

Author

Rubén Sánchez Castellano

Date

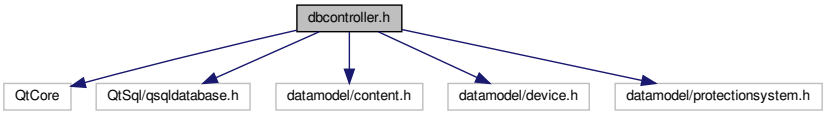
August 24, 2018

5.3 dbcontroller.h File Reference

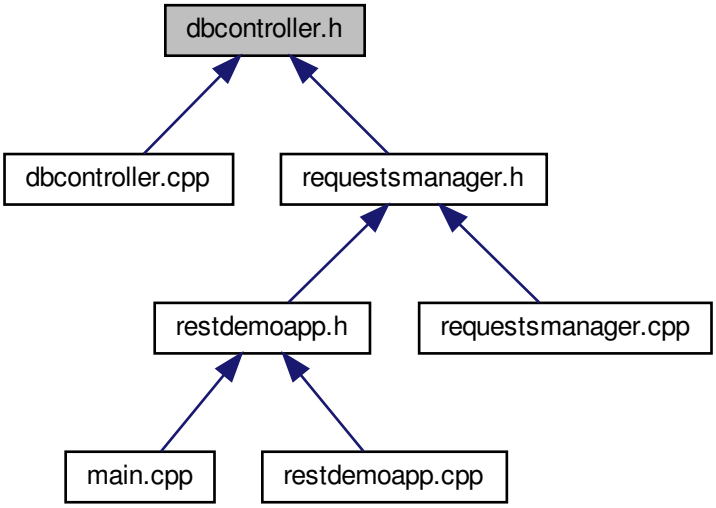
Database Controller class declaration.

```
#include <QtCore>
#include <QtSql/qsqldatabase.h>
#include "datamodel/content.h"
#include "datamodel/device.h"
#include "datamodel/protectionsystem.h"
```

Include dependency graph for dbcontroller.h:



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



Classes

- class DBController

  The DBController class implements the API to handle an SQLite database using the native Qt API for SQL databases. This controller only provides methods to work with content data as specified on the requeriments. This class des not implements any business logic.

5.3.1 Detailed Description

Database Controller class declaration.

Author

Rubén Sánchez Castellano

Date

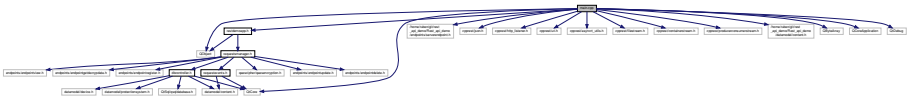
August 24, 2018

5.4 main.cpp File Reference

Main file.

```
#include "restdemoapp.h"
#include <QCoreApplication>
#include <QtDebug>
```

Include dependency graph for main.cpp:



Macros

- #define **RUN_MODE_NORMAL** 0
- #define **RUN_MODE_UNIT_TEST** 1

Functions

- int **main** (int argc, char ∗argv[ ])

5.4.1 Detailed Description

Main file.

Author

Rubén Sánchez Castellano

Date

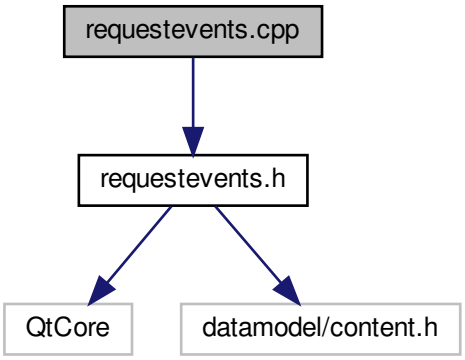
August 24, 2018

5.5 requestevents.cpp File Reference

Request events message id registration on Qt.

```
#include "requestevents.h"
```

Include dependency graph for requestevents.cpp:



5.5.1 Detailed Description

Request events message id registration on Qt.

Author

Rubén Sánchez Castellano

Date

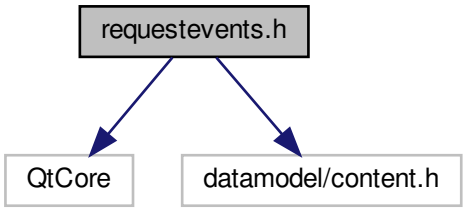
August 24, 2018

5.6 requestevents.h File Reference

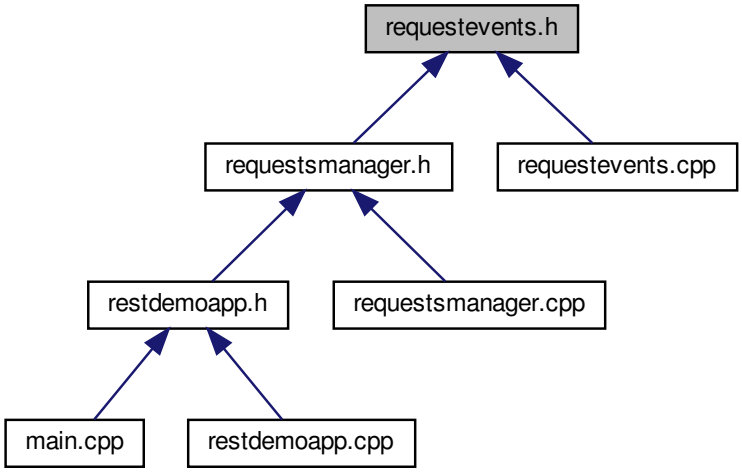
Requests events declaration. These events are used by the endpoints to notify the application the requests received and their parameters. These messages inherit from the Qt native QEvent class and they're delivered to the application using the Qt's native message system on an asynchronous way. By this way we can ensure when a request is parsed, the endpoints thread goes back to listening for new requests since the message is then handled by another thread (Qt's Event Loop thread).

```
#include <QtCore>
#include "datamodel/content.h"
```

Include dependency graph for requestevents.h:



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



Classes

- class RegisterRequestEvent

  The RegisterRequestEvent class declares the message to notify a new Register request.
- class UpdateRequestEvent

  The UpdateRequestEvent class declares the message to notify a new Update request.
- class DeleteRequestEvent

  The DeleteRequestEvent class declares the message to notify a new Delete request.
- class ViewRequestEvent

  The ViewRequestEvent class declares the message to notify a new View request.
- class GetDecryptDataRequestEvent

  The GetDecryptDataRequestEvent class declares the message to notify a new request to get some content data decrypted.

5.6.1 Detailed Description

Requests events declaration. These events are used by the endpoints to notify the application the requests received and their parameters. These messages inherit from the Qt native QEvent class and they're delivered to the application using the Qt's native message system on an asynchronous way. By this way we can ensure when a request is parsed, the endpoints thread goes back to listening for new requests since the message is then handled by another thread (Qt's Event Loop thread).

Author

Rubén Sánchez Castellano

Date

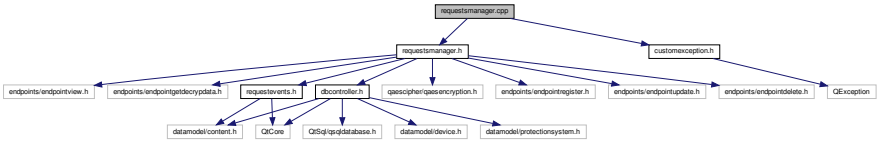
August 24, 2018

5.7 requestsmanager.cpp File Reference

Request Manager class definition.

```
#include "requestsmanager.h"
#include "customexception.h"
```

Include dependency graph for requestsmanager.cpp:



5.7.1 Detailed Description

Request Manager class definition.

Author

Rubén Sánchez Castellano

Date

August 24, 2018

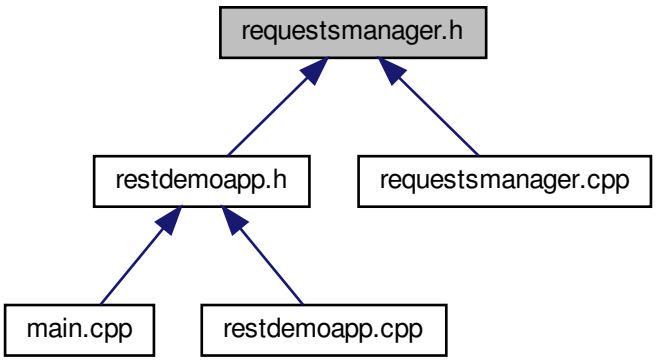
5.8 requestsmanager.h File Reference

Requests Manager class declaration.

```
#include "endpoints/endpointregister.h"
#include "endpoints/endpointupdate.h"
#include "endpoints/endpointdelete.h"
#include "endpoints/endpointview.h"
#include "endpoints/endpointgetdecrypdata.h"
#include "dbcontroller.h"
#include "requestevents.h"
#include "qaescipher/qaesencryption.h"
```

Include dependency graph for requestsmanager.h:

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



Classes

- class RequestsManager

  The RequestsManager class implements the business logic of the application. It recieves the requests notified by the endpoints and uses the Database Controller to handle the persistent data.

5.8.1 Detailed Description

Requests Manager class declaration.

Author

Rubén Sánchez Castellano

Date

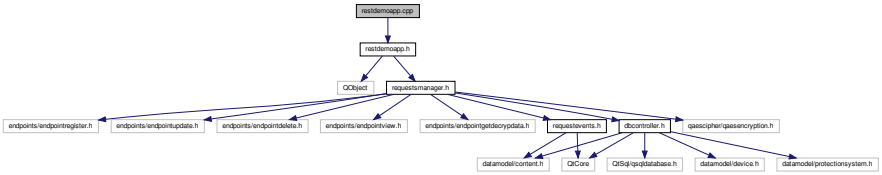
August 24, 2018

5.9 restdemoapp.cpp File Reference

RestDemoApp class definition.

```
#include "restdemoapp.h"
```

Include dependency graph for restdemoapp.cpp:



5.9.1 Detailed Description

RestDemoApp class definition.

Author

Rubén Sánchez Castellano

Date

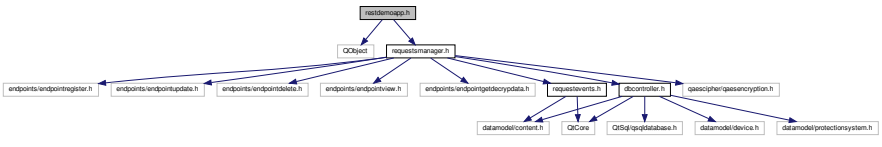
August 24, 2018

5.10 restdemoapp.h File Reference

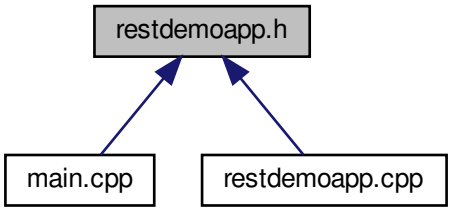
RestDemoApp class declaration.

```
#include <QObject>
#include "requestsmanager.h"
```

Include dependency graph for restdemoapp.h:



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



Classes

- class RestDemoApp

  The RestDemoApp class implements the initialization of the application. In addition, this class receives the messages from the endpoints notifying new requests and delegates them to the Requests Manager. So, this class does not implements any business logic.

5.10.1 Detailed Description

RestDemoApp class declaration.

Author

Rubén Sánchez Castellano

Date

August 24, 2018

Index