SQL Backup to FTP server

Generated by Doxygen 1.9.3

Chapter 1

SQL Backup to FTP server

1.1 Introduction

The app was written by using **C++17** standart.
Tests are written by using **Google Test** framework.
The app uses **sqlite3** database and **Filezilla** FTP server.

App command line options:
–ftp_host' - Flag to specify the backup FTP server's host.
–ftp_port - Flag to specify the backup FTP server's port.
–ftp_uname - Flag to specify the backup FTP server's User name.
–ftp_pass - Flag to specify the backup FTP server's User password.
–ftp_dir - Flag to specify the backup FTP server's copy directory.

sqlite3 doesn't require a **connection properties** so I didn't add an option to add a db property.

1.2 How to use

1.2.1 Step 1: Set up an Filezilla FTP server and create a user in it.

1.2.2 Step 2: Run the program by setting options via the console.

example: **./app –ftp_host 127.0.0.1 –ftp_port 21 –ftp_uname username –ftp_passwd password –ftp_dir ftpDir**

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 ftp Namespace Reference

Namespaces

- namespace **dbmgr**
- namespace **query**

Classes

- class **CmdLineOptionParser**
- class **FTPManager**
- class **Logger**
- class **SQLBackup**

Functions

- std::string **getDateTime** ()

5.1.1 Function Documentation

5.1.1.1 getDateTime()

```
std::string ftp::getDateTime ( )
```

5.2 ftp::dbmgr Namespace Reference

Classes

- class **TableData**

Functions

- bool **init** (const std::string &dbPath, std::string &errorMsg)
- **TableData** **selectData** (const std::string &query, std::string &errorMsg)
- bool **removeDb** (const std::string &dbPath, std::string &errorMsg)
- bool **insertData** (const std::string &query, std::string &errorMsg)
- bool **createTable** (const std::string &query, std::string &errorMsg)

5.2.1 Function Documentation

5.2.1.1 createTable()

```
bool ftp::dbmgr::createTable (
            const std::string & query,
            std::string & errorMsg )
```

Creates a table in the database according to the given query.

Parameters

errorMsg	if an error occurs then the error text is assigned to this variable

Returns

true if there is no errors

5.2.1.2 init()

```
bool ftp::dbmgr::init (
            const std::string & dbPath,
            std::string & errorMsg )
```

Creates a database at the given path.

Parameters

dbPath	path to database
errorMsg	if an error occurs then the error text is assigned to this variable

Returns

true if there is no errors

5.2.1.3 insertData()

```
bool ftp::dbmgr::insertData (
             const std::string & query,
             std::string & errorMsg )
```

Adds data to the database according to the given query.

Parameters

errorMsg	if an error occurs then the error text is assigned to this variable

Returns

true if there is no errors

5.2.1.4 removeDb()

```
bool ftp::dbmgr::removeDb (
             const std::string & dbPath,
             std::string & errorMsg )
```

Removes a database at the given path.

Parameters

dbPath	path to database
errorMsg	if an error occurs then the error text is assigned to this variable

Returns

true if there is no errors

5.2.1.5 selectData()

```
 TableData ftp::dbmgr::selectData (
             const std::string & query,
             std::string & errorMsg )
```

Fetches data from the database

Parameters

errorMsg	if an error occurs then the error text is assigned to this variable

5.3 ftp::query Namespace Reference

Variables

- const std::string **createTable**
- const std::string **insertData**
- const std::string **selectData** = "SELECT ∗ FROM Person;"

5.3.1 Variable Documentation

5.3.1.1 createTable

```
const std::string ftp::query::createTable
```

Initial value:

```
=
        "CREATE TABLE IF NOT EXISTS Person("
        "ID INT PRIMARY KEY     NOT NULL, "
        "NAME           VARCHAR NOT NULL, "
        "AGE            INT     NOT NULL, "
        "ADDRESS        VARCHAR, "
        "SALARY         REAL );"
```

5.3.1.2 insertData

```
const std::string ftp::query::insertData
```

Initial value:

```
=
        "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (1, 'AAAA', 11, 'Armenia', 10000 ); "
        "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (2, 'BBBB', 22, 'Germany', 20000 ); "
        "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (3, 'CCCC', 33, 'France', 30000 ); "
        "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (4, 'DDDD', 44, 'USA', 40000 ); "
        "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (5, 'EEEE', 55, 'UK', 500000 ); "
```

5.3.1.3 selectData

```
const std::string ftp::query::selectData = "SELECT * FROM Person;"
```

Chapter 6

Class Documentation

6.1 ftp::CmdLineOptionParser Class Reference

```
#include <CmdLineOptionParser.h>
```

Public Member Functions

- void **parseArgs** (int argc, char ∗argv[ ])
- bool **checkOption** (const std::string &option) const
- void **addOption** (const std::string &option, const std::string &description)
- int **getIntOptionValue** (const std::string &option) const
- std::string **getStringOptionValue** (const std::string &option) const
- void **printHelp** () const

6.1.1 Detailed Description

Class to parse the application's command line options.

6.1.2 Member Function Documentation

6.1.2.1 addOption()

```
void ftp::CmdLineOptionParser::addOption (
             const std::string & option,
             const std::string & description )
```

Adds an option and its description to the map.

6.1.2.2 checkOption()

```
bool ftp::CmdLineOptionParser::checkOption (
             const std::string & option ) const
```

Checks if the given option is in the map.

Returns

True if option found

6.1.2.3 getIntOptionValue()

```
int ftp::CmdLineOptionParser::getIntOptionValue (
             const std::string & option ) const
```

Returns the value for the given key as an integr.

Exceptions

runtime_exception	when option not found

6.1.2.4 getStringOptionValue()

```
std::string ftp::CmdLineOptionParser::getStringOptionValue (
             const std::string & option ) const
```

Returns the value for the given key as an string.

Exceptions

runtime_exception	when option not found

6.1.2.5 parseArgs()

```
void ftp::CmdLineOptionParser::parseArgs (
             int argc,
             char * argv[] )
```

Parses the arguments and their values and stores them in the map.

Parameters

argc	argument count
argv	argument list

6.1.2.6 printHelp()

```
void ftp::CmdLineOptionParser::printHelp ( ) const
```

Prints help to the console.

6.2 ftp::FTPManager Class Reference

```
#include <FTPManager.h>
```

Public Member Functions

- **FTPManager** (int port, const std::string &host, const std::string &uname, const std::string &passwd, const std::string &dirPath)
- ∼**FTPManager** ()
- void **endSession** ()
- bool **startSession** ()
- std::string **getErrorMsg** () const
- bool **transfer** (const std::string &localfilePath, const std::string &newRemotefile)

6.2.1 Detailed Description

Class to manage file transfer between Application and FTP Server.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 FTPManager()

```
ftp::FTPManager::FTPManager (
            int port,
            const std::string & host,
            const std::string & uname,
            const std::string & passwd,
            const std::string & dirPath )
```

6.2.2.2 ∼FTPManager()

```
ftp::FTPManager::∼FTPManager ( )
```

6.2.3 Member Function Documentation

6.2.3.1 endSession()

```
void ftp::FTPManager::endSession ( )
```

Disconnects from the FTP server and from the session.

6.2.3.2 getErrorMsg()

```
std::string ftp::FTPManager::getErrorMsg ( ) const
```

If one of the functions does not work correctly and there will be an error, then each function returns false and assigns the current error to the errorMsg_ variable.

Returns

the current error message

6.2.3.3 startSession()

```
bool ftp::FTPManager::startSession ( )
```

Establishes a connection with the FTP server and creates a directory dirPath _ if not already created and starts a new session.

Returns

true if everything is ok

6.2.3.4 transfer()

```
bool ftp::FTPManager::transfer (
             const std::string & localfilePath,
             const std::string & newRemotefile )
```

Sends a file to specifed FTP Server directory.

Parameters

localfilePath	local absolute path of a file
newremotefile	new name of a file

Returns

true if transfer was succsessful

6.3 ftp::Logger Class Reference

```
#include <Logger.h>
```

Static Public Member Functions

- static void **error** (std::string_view fileName, std::string_view msg)
- static void **info** (std::string_view fileName, std::string_view msg)

6.3.1 Detailed Description

Siple logger.

6.3.2 Member Function Documentation

6.3.2.1 error()

```
static void ftp::Logger::error (
             std::string_view fileName,
             std::string_view msg )  [inline], [static]
```

6.3.2.2 info()

```
static void ftp::Logger::info (
             std::string_view fileName,
             std::string_view msg )  [inline], [static]
```

6.4 ftp::SQLBackup Class Reference

```
#include <SQLBackup.h>
```

Public Member Functions

- **SQLBackup** ()
- void **start** (int argc, char ∗argv[ ])

6.4.1 Detailed Description

Class to manage the entire workflow in a project.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 SQLBackup()

```
ftp::SQLBackup::SQLBackup ( )
```

6.4.3 Member Function Documentation

6.4.3.1 start()

```
void ftp::SQLBackup::start (
              int argc,
              char * argv[ ] )
```

Creates data for a SQLite database, and then exports the database data to a file and sends the file to an FTP server.

Parameters

argc	argument count
argv	argument list

6.5 ftp::dbmgr::TableData Class Reference

```
#include <DbManager.h>
```

Public Member Functions

- void **print** ()
- void **dumpToFile** (const std::string &filePath)
- bool **isEmptty** () const

Public Attributes

- std::vector< std::vector< std::string > > **data**

6.5.1 Detailed Description

Class container which holds the data that is returned from db.

6.5.2 Member Function Documentation

6.5.2.1 dumpToFile()

```
void ftp::dbmgr::TableData::dumpToFile (
             const std::string & filePath )
```

6.5.2.2 isEmptty()

```
bool ftp::dbmgr::TableData::isEmptty ( ) const
```

6.5.2.3 print()

```
void ftp::dbmgr::TableData::print ( )
```

6.5.3 Member Data Documentation

6.5.3.1 data

```
std::vector<std::vector<std::string> > ftp::dbmgr::TableData::data
```

Chapter 7

File Documentation

7.1 CmdLineOptionParser.cpp File Reference

```
#include "CmdLineOptionParser.h"
#include "Logger.h"
#include <iostream>
#include <string.h>
#include <filesystem>
```

Namespaces

- namespace **ftp**

7.2 CmdLineOptionParser.h File Reference

```
#include <unordered_map>
```

Classes

- class **ftp::CmdLineOptionParser**

Namespaces

- namespace **ftp**

7.3 CmdLineOptionParser.h

Go to the documentation of this file.

```
#ifndef FTP_CMDLINEOPTIONPARSER_H_
#define FTP_CMDLINEOPTIONPARSER_H_

#include <unordered_map>

namespace ftp {

    class CmdLineOptionParser
    {
        using HashContainerType = std::unordered_map<std::string, std::string>;
    public:

        void parseArgs(int argc, char* argv[]);

        bool checkOption(const std::string& option) const;

        void addOption(const std::string& option, const std::string& description);

        int getIntOptionValue(const std::string& option) const;

        std::string getStringOptionValue(const std::string& option) const;

        void printHelp() const;

    private:
        HashContainerType optionToValueMap_;
        HashContainerType optionToDescriptionMap_;
    };

} // end namespace ftp
#endif // FTP_CMDLINEOPTIONPARSER_H_
```

7.4 DbManager.cpp File Reference

```
#include "DbManager.h"
#include "Logger.h"
#include <fstream>
#include <filesystem>
#include <winsqlite/winsqlite3.h>
```

Namespaces

- namespace **ftp**
- namespace **ftp::dbmgr**

Functions

- bool **ftp::dbmgr::init** (const std::string &dbPath, std::string &errorMsg)
- TableData **ftp::dbmgr::selectData** (const std::string &query, std::string &errorMsg)
- bool **ftp::dbmgr::removeDb** (const std::string &dbPath, std::string &errorMsg)
- bool **ftp::dbmgr::insertData** (const std::string &query, std::string &errorMsg)
- bool **ftp::dbmgr::createTable** (const std::string &query, std::string &errorMsg)

7.5 DbManager.h File Reference

```
#include <string>
#include <vector>
#include <sqlite3.h>
#include <filesystem>
```

Classes

- class **ftp::dbmgr::TableData**

Namespaces

- namespace **ftp**
- namespace **ftp::dbmgr**

Functions

- bool **ftp::dbmgr::init** (const std::string &dbPath, std::string &errorMsg)
- bool **ftp::dbmgr::insertData** (const std::string &query, std::string &errorMsg)
- bool **ftp::dbmgr::createTable** (const std::string &query, std::string &errorMsg)
- TableData **ftp::dbmgr::selectData** (const std::string &query, std::string &errorMsg)
- bool **ftp::dbmgr::removeDb** (const std::string &dbPath, std::string &errorMsg)

7.6 DbManager.h

Go to the documentation of this file.

```
#ifndef FTP_DBMANAGER_H_
#define FTP_DBMANAGER_H_

#include <string>
#include <vector>
#include <sqlite3.h>
#include <filesystem>

namespace ftp::dbmgr {

    class TableData
    {
    public:
        void print();
        void dumpToFile(const std::string& filePath);
        bool isEmptty() const;

        std::vector<std::vector<std::string» data;
    };

    bool init(const std::string& dbPath, std::string& errorMsg);

    bool insertData(const std::string& query, std::string& errorMsg);

    bool createTable(const std::string& query, std::string& errorMsg);

    TableData selectData(const std::string& query, std::string& errorMsg);

    bool removeDb(const std::string& dbPath, std::string& errorMsg);

} // end namespace dbmgr
#endif // FTP_LOGGER_H_
```

7.7 FTPManager.cpp File Reference

```
#include "FTPManager.h"
#include "Logger.h"
```

Namespaces

- namespace **ftp**

7.8 FTPManager.h File Reference

```
#include <string>
#include <Windows.h>
#include <WinInet.h>
```

Classes

- class **ftp::FTPManager**

Namespaces

- namespace **ftp**

7.9 FTPManager.h

Go to the documentation of this file.

```
#ifndef FTP_FTPMANAGER_H_
#define FTP_FTPMANAGER_H_

#include <string>
#include <Windows.h>
#include <WinInet.h>

namespace ftp {

    class FTPManager
    {
    public:
        FTPManager(int port, const std::string& host, const std::string& uname,
            const std::string& passwd, const std::string& dirPath);

        ~FTPManager();

        void endSession();

        bool startSession();

        std::string getErrorMsg() const;

        bool transfer(const std::string& localfilePath, const std::string& newRemotefile);

    private:
        bool setCurrentDir(const std::string& path);
        bool createDirIfNotExists(const std::string& path);

        int port_;
        std::string host_;
        std::string uname_;
        std::string passwd_;
        std::string dirPath_;
        std::string errorMsg_;

        HINTERNET session_ = nullptr;
        HINTERNET internet_ = nullptr;
    };

} // end namespace ftp
#endif // FTP_FTPMANAGER_H_
```

7.10 Logger.h File Reference

```
#include <iostream>
#include <string_view>
```

Classes

- class **ftp::Logger**

Namespaces

- namespace **ftp**

7.11 Logger.h

Go to the documentation of this file.

```
#ifndef FTP_LOGGER_H_
#define FTP_LOGGER_H_

#include <iostream>
#include <string_view>

namespace ftp {

    class Logger
    {
    public:
        static void error(std::string_view fileName, std::string_view msg)
        {
            std::cerr « "ERROR: in " « fileName « ": " « msg « std::endl;
        }

        static void info(std::string_view fileName, std::string_view msg)
        {
            std::cout « "INFO: " « fileName « ": " « msg « std::endl;
        }

    private:
        Logger() = delete;
        ~Logger() = delete;
        Logger(const Logger&) = delete;
        void operator=(const Logger&) = delete;
        Logger(Logger&&) noexcept = delete;
        Logger& operator=(Logger&&) noexcept = delete;
    };

} // end namespace ftp
#endif // FTP_LOGGER_H_
```

7.12 main.cpp File Reference

```
#include "SQLBackup.h"
```

Functions

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

7.12.1 Function Documentation

7.12.1.1 main()

```
int main (
            int argc,
            char * argv[] )
```

7.13 QueryCollection.h File Reference

```
#include <string>
```

Namespaces

- namespace **ftp**
- namespace **ftp::query**

Variables

- const std::string **ftp::query::createTable**
- const std::string **ftp::query::insertData**
- const std::string **ftp::query::selectData** = "SELECT ∗ FROM Person;"

7.14 QueryCollection.h

Go to the documentation of this file.

```
#ifndef FTP_QUERYCOLLECTION_H_
#define FTP_QUERYCOLLECTION_H_

#include <string>

namespace ftp::query {

    const std::string createTable =
        "CREATE TABLE IF NOT EXISTS Person("
         "ID INT PRIMARY KEY     NOT NULL, "
         "NAME           VARCHAR NOT NULL, "
         "AGE            INT     NOT NULL, "
         "ADDRESS        VARCHAR, "
         "SALARY         REAL );";

     const std::string insertData =
         "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (1, 'AAAA', 11, 'Armenia', 10000 ); "
         "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (2, 'BBBB', 22, 'Germany', 20000 ); "
         "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (3, 'CCCC', 33, 'France', 30000 ); "
         "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (4, 'DDDD', 44, 'USA', 40000 ); "
         "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (5, 'EEEE', 55, 'UK', 500000 ); ";

     const std::string selectData = "SELECT * FROM Person;";

 }// end namespace ftp::query
 #endif // FTP_QUERYCOLLECTION_H_
```

7.15 SQLBackup.cpp File Reference

```
#include "SQLBackup.h"
#include "Logger.h"
#include "DbManager.h"
#include "FTPManager.h"
#include "QueryCollection.h"
#include <ctime>
#include <fstream>
#include <stdexcept>
#include <filesystem>
```

Namespaces

- namespace **ftp**

Functions

- std::string **ftp::getDateTime** ()

7.16 SQLBackup.h File Reference

```
#include "CmdLineOptionParser.h"
```

Classes

- class **ftp::SQLBackup**

Namespaces

- namespace **ftp**

7.17 SQLBackup.h

Go to the documentation of this file.

```
#ifndef SQLBACKUP_H_
#define SQLBACKUP_H_

#include "CmdLineOptionParser.h"

namespace ftp {

    class SQLBackup
    {
    public:
        SQLBackup();

        void start(int argc, char* argv[]);

    private:
        void initArgParser();

        CmdLineOptionParser cmdParser_;
    };

} // end namespace ftp
#endif // SQLBACKUP_H_
```

Index