Focus

1.0

Generated by Doxygen 1.8.17

Chapter 1

Focus

1.1 Project

Focus is a messaging application tailored for workers, students or those with group projects. Carefully designed to aid project development and planning giving users the ability to effectively communicate with eachother. Through the use of roles and permissions and customization features, Focus gives a unique and tailored experience, a perfect tool for your workspace.

1.2 References

To achieve the results shown in this application this external tools were used:

- Flaticon
- Doxygen
- Qt
- Mqtt

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

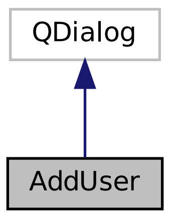
Class Documentation

4.1 AddUser Class Reference

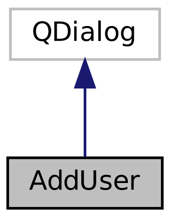
The AddUser class.

```
#include <adduser.h>
```

Inheritance diagram for AddUser:



Collaboration diagram for AddUser:



Public Member Functions

- AddUser (SQL ∗sql, User ∗user, QWidget ∗parent=nullptr)

  Constructor for AddUser class.

4.1.1 Detailed Description

The AddUser class.

This class is used to generate the AddUser Dialog window

4.1.2 Constructor & Destructor Documentation

4.1.2.1 AddUser()

```
AddUser::AddUser (
             SQL * sql,
             User * user,
             QWidget * parent = nullptr )  [explicit]
```

Constructor for AddUser class.

Parameters

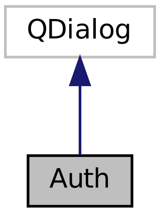
sql	
user	
parent	

4.2 Auth Class Reference

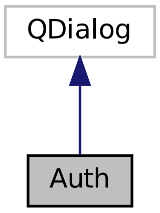
The Auth class.

```
#include <auth.h>
```

Inheritance diagram for Auth:



Collaboration diagram for Auth:



Public Member Functions

- Auth (SQL ∗sql, QWidget ∗parent=nullptr)

  Constructor for Auth.
- QString validateLogin (QString email, QString password, QString encryptedPassword)

  validateLogin - validates the credentials provided by the user and checks if it is a valid login
- QString validateRegister (User ∗user, QString password, QString confirmPassword)

  validateRegister - validates the information provided by the user

4.2.1 Detailed Description

The Auth class.

This class is used to create the authentication page

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Auth()

```
Auth::Auth (
            SQL * sql,
            QWidget * parent = nullptr )  [explicit]
```

Constructor for Auth.

Parameters

sql	
parent	

4.2.3 Member Function Documentation

4.2.3.1 validateLogin()

```
QString Auth::validateLogin (
            QString email,
            QString password,
            QString encryptedPassword )
```

validateLogin - validates the credentials provided by the user and checks if it is a valid login

Parameters

email	
password	
encryptedPassword	

Returns

4.2.3.2 validateRegister()

```
QString Auth::validateRegister (
            User * user,
            QString password,
            QString confirmPassword )
```

validateRegister - validates the information provided by the user

Parameters

user	
password	
confirmPassword	

Returns

4.3 Chatroom Class Reference

The Chatroom class.

```
#include <chatroom.h>
```

Public Member Functions

- Chatroom (QString name, QVector< User > members, QString topic)

    Constructor for Chatroom class.
- QString **name** () const
- void **setName** (const QString &name)
- QString **topic** () const
- void **setTopic** (const QString &topic)
- QVector< User > **members** () const
- void **setMembers** (const QVector< User > &members)

4.3.1 Detailed Description

The Chatroom class.

This class is used to store the chatroom information

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Chatroom()

```
Chatroom::Chatroom (
            QString name,
            QVector< User > members,
            QString topic )
```

Constructor for Chatroom class.

Parameters

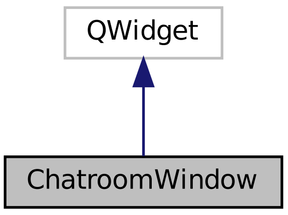
name	
members	
topic	

4.4 ChatroomWindow Class Reference

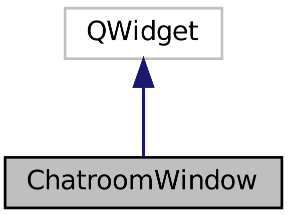
The Chatroom class.

`#include <chatroomwindow.h>`

Inheritance diagram for ChatroomWindow:



Collaboration diagram for ChatroomWindow:



Public Member Functions

- ChatroomWindow (Chatroom chatroom, Mosquitto ∗mainClient, SQL ∗sql, User ∗user, QWidget ∗parent=nullptr)

  Constructor for ChatroomWindow class.
- void loadMessages ()

loadMessages - loads the messages for the respective chatroom

- void loadMembers ()

  loadMembers - loads the members for the respective chatroom
- void loadChatroom (Chatroom chatroom)

  loadChatroom - loads the chatroom information
- void **onMessageReceived** (const QMqttMessage &msg)

4.4.1 Detailed Description

The Chatroom class.

This class is used to create the chatroom windows with QWidget

4.4.2 Constructor & Destructor Documentation

4.4.2.1 ChatroomWindow()

```
ChatroomWindow::ChatroomWindow (
             Chatroom chatroom,
             Mosquitto * mainClient,
             SQL * sql,
             User * user,
             QWidget * parent = nullptr )  [explicit]
```

Constructor for ChatroomWindow class.

Parameters

chatroom	
mainClient	
sql	
user	
parent	

4.4.3 Member Function Documentation

4.4.3.1 loadChatroom()

```
void ChatroomWindow::loadChatroom (
             Chatroom chatroom )
```

loadChatroom - loads the chatroom information

Parameters

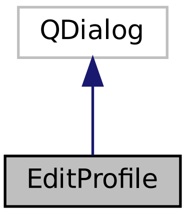
chatroom	

4.5 EditProfile Class Reference

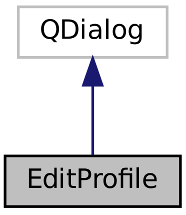
The EditProfile class.

```
#include <editprofile.h>
```

Inheritance diagram for EditProfile:



Collaboration diagram for EditProfile:



Public Member Functions

- EditProfile (SQL ∗sql, User ∗user, QWidget ∗parent=nullptr)

    Constructor for EditProfile class.

4.5.1 Detailed Description

The EditProfile class.

This class is used to create the EditProfile dialog window

4.5.2 Constructor & Destructor Documentation

4.5.2.1 EditProfile()

```
EditProfile::EditProfile (
            SQL * sql,
            User * user,
            QWidget * parent = nullptr )  [explicit]
```

Constructor for EditProfile class.

Parameters

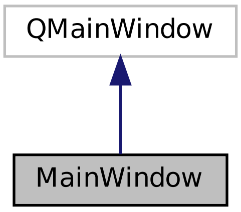
user	
parent	

4.6 MainWindow Class Reference

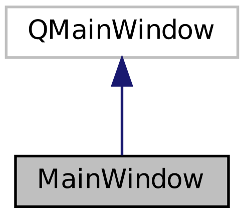
The MainWindow class.

```
#include <mainwindow.h>
```

Inheritance diagram for MainWindow:



Collaboration diagram for MainWindow:



Public Member Functions

- MainWindow (SQL ∗sql, Mosquitto ∗mainClient, User ∗user, QWidget ∗parent=nullptr)

    Constructor for MainWindow class.
- void newChatroomWindow (Chatroom chatroom)

    newChatroomWindow - creates new ChatroomWindow Widgets
- void loadGroupsList ()

    loadGroupsList - loads the user's groups
- void loadDmsList ()

    loadDmsList - loads the user's direct messages
- void loadChatrooms ()

    loadChatrooms - loads all the Chatroom classes (groups and dms)

4.6.1 Detailed Description

The MainWindow class.

This class is used to create the application's main window page using QMainWindow

4.6.2 Constructor & Destructor Documentation

4.6.2.1 MainWindow()

```
MainWindow::MainWindow (
             SQL ∗ sql,
             Mosquitto ∗ mainClient,
             User ∗ user,
             QWidget ∗ parent = nullptr )
```

Constructor for MainWindow class.

Parameters

sql	
mainClient	
user	
parent	

Note

Workaround because mosquitto failed when loading from the constructor

4.6.3 Member Function Documentation

4.6.3.1 newChatroomWindow()

```
void MainWindow::newChatroomWindow (
            Chatroom chatroom )
```

newChatroomWindow - creates new ChatroomWindow Widgets

Parameters

chatroom	

4.7 Message Class Reference

The Message class.

```
#include <message.h>
```

Public Member Functions

- Message (QString content, QString dateString, QString topic, QString username)

  Constructor for Message class.
- QString **content** () const
- void **setContent** (const QString &content)
- QString **topic** () const
- void **setTopic** (const QString &topic)
- QString **username** () const
- void **setUsername** (const QString &username)
- QDateTime **dateTime** () const
- void **setDateTime** (const QDateTime &dateTime)

4.7.1 Detailed Description

The Message class.

This class manages the messages' information

4.7.2 Constructor & Destructor Documentation

4.7.2.1 Message()

```
Message::Message (
            QString content,
            QString dateString,
            QString topic,
            QString username )
```

Constructor for Message class.

Parameters

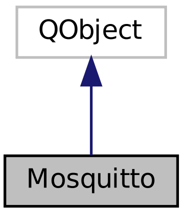
content	
dateString	
topic	
username	

4.8 Mosquitto Class Reference

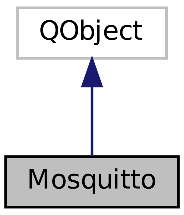
The Mosquitto class.

```
#include <mosquitto.h>
```

Inheritance diagram for Mosquitto:



Collaboration diagram for Mosquitto:



Public Member Functions

- Mosquitto (QString host, int port, QString username)

  Constructor for Mosquitto class.
- QMqttClient ∗ **client** () const
- void publish (QString message, QString topic)

  publish - publishes a message in a specific topic with mqtt
- QMqttSubscription ∗ subscribe (QString topic)

  subscribe - subscribes to a topic with mqtt

4.8.1 Detailed Description

The Mosquitto class.

This class is used to process mqtt's features

4.8.2 Constructor & Destructor Documentation

4.8.2.1 Mosquitto()

```
Mosquitto::Mosquitto (
             QString host,
             int port,
             QString username )
```

Constructor for Mosquitto class.

Parameters

port	
username	

4.8.3 Member Function Documentation

4.8.3.1 publish()

```
void Mosquitto::publish (
             QString message,
             QString topic )
```

publish - publishes a message in a specific topic with mqtt

Parameters

message	
topic	

4.8.3.2 subscribe()

```
QMqttSubscription * Mosquitto::subscribe (
             QString topic )
```

subscribe - subscribes to a topic with mqtt

Parameters

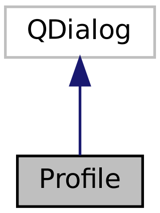
topic	

4.9 Profile Class Reference

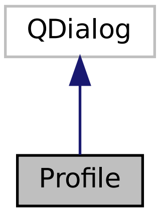
The Profile class.

```
#include <profile.h>
```

Inheritance diagram for Profile:



Collaboration diagram for Profile:



Public Member Functions

- Profile (SQL ∗sql, User ∗currentUser, User ∗user, QWidget ∗parent=nullptr)

  Constructor for Profile class.

4.9.1 Detailed Description

The Profile class.

This class is used to create the profile dialog window

4.9.2 Constructor & Destructor Documentation

4.9.2.1 Profile()

```
Profile::Profile (
            SQL * sql,
            User * currentUser,
            User * user,
            QWidget * parent = nullptr )  [explicit]
```

Constructor for Profile class.

Parameters

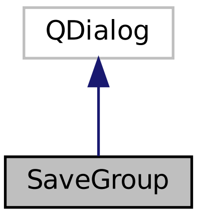
sql	
currentUser	
user	
parent	

4.10 SaveGroup Class Reference

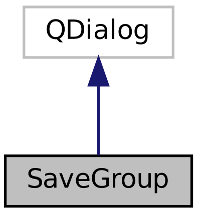
The Profile class.

`#include <savegroup.h>`

Inheritance diagram for SaveGroup:



Collaboration diagram for SaveGroup:



Public Member Functions

- SaveGroup (SQL ∗sql, User ∗user, Chatroom ∗group=nullptr, QWidget ∗parent=nullptr)

  Constructor for SaveGroup class.
- void createNewGroup ()

  createNewGroup - creates a new group
- void updateGroup ()

  updateGroup - updates the group's information
- void loadContactsList ()

  loadContactsList - loads the user's contacts
- void loadMembersList ()

  loadMembersList - loads the group's members
- Chatroom ∗ **group** () const
- void **setGroup** (Chatroom ∗group)

Public Attributes

- bool **_deleted** = false

4.10.1 Detailed Description

The Profile class.

This class is used to create the SaveGroup dialog window

4.10.2 Constructor & Destructor Documentation

4.10.2.1 SaveGroup()

```
SaveGroup::SaveGroup (
             SQL * sql,
             User * user,
             Chatroom * group = nullptr,
             QWidget * parent = nullptr )   [explicit]
```

Constructor for SaveGroup class.

Parameters

sql	
user	
group	
parent	

4.11 SQL Class Reference

The SQL class.

```
#include <sql.h>
```

Public Member Functions

- SQL ()

  Constructor for SQL class.
- QString validateTopic (QString topic)

  validateTopic - validates the topic name
- void addGroup (QString name, QString topic, User user)

  addGroup - adds a new group to the database
- void addUser (User ∗user, QString passwordHash)

  addUser - adds a new user to the database
- void addContactByUsername (User ∗user, QString contactUsername)

  addContactByUsername - connects the user to the requested contact
- int **addMessage** (Message &message)
- void addSubscription (QString topic, User ∗user)

  addSubscription - stores subscription in database
- void updateGroup (Chatroom ∗group)

  updateGroup - updates the group information
- void updateUser (User ∗user)

  updateUser - updates the user information
- void removeSubscriptions (QString topic)

  removeSubscriptions - removes the stored subscription from the database
- void removeContact (User ∗currentUser, User ∗user)

  removeContact - removes the contact for respective user
- void removeGroup (Chatroom ∗group)

  removeGroup - removes group information from database
- QVector< QString > getSubscriptionsByUser (User ∗user)

  getSubscriptionsByUser - returns subscriptions for respective user
- QVector< Message > getMessagesByTopic (QString topic, int limit=30)

  getMessagesByTopic - returns the last 30 messages for respective topic
- QVector< User > getUserContacts (User ∗user)

  getUserContacts - returns vector with the user's contacts
- QVector< Chatroom > getUserDms (User ∗user)

  getUserDms - returns the direct messages of the respective user
- QVector< Chatroom > getGroupsByTopic (QString topic)

  getGroupsByTopic - returns the repective user's groups
- QVector< User > getMembersByTopic (QString topic)

  getMembersByTopic - returns the members subscribe to the respective topic
- User ∗ getUserByUsername (QString username)

  getUserByUsername - returns the user's information by username
- User ∗ getUserByEmail (QString email)

  getUserByEmail - returns the user's information by email
- Message ∗ getMessageById (QString id)

  getMessageById - returns the message information by id
- bool isValidLoginByEmail (QString email, QString passwordHash)

isValidLoginByEmail - checks if the login by email is valid

- bool isValidLoginByUsername (QString username, QString passwordHash)

  isValidLoginByUsername - checks if the login by username is valid
- bool isGroupAdmin (User ∗user, QString topic)

  isGroupAdmin - returns true if the respective user is the group's admin
- bool areContacts (User ∗currentUser, User ∗user)

  areContacts - checks if the users are connected

4.11.1 Detailed Description

The SQL class.

This class is used to process mysql features

4.11.2 Member Function Documentation

4.11.2.1 addContactByUsername()

```
void SQL::addContactByUsername (
            User * user,
            QString contactUsername )
```

addContactByUsername - connects the user to the requested contact

Parameters

user	
contactUsername	

Returns

bool (true if successfull)

4.11.2.2 addGroup()

```
void SQL::addGroup (
            QString name,
            QString topic,
            User user )
```

addGroup - adds a new group to the database

Parameters

name	
topic	
user	

4.11.2.3 addSubscription()

```
void SQL::addSubscription (
             QString topic,
             User * user )
```

addSubscription - stores subscription in database

Parameters

topic	
user	

4.11.2.4 addUser()

```
void SQL::addUser (
             User * user,
             QString passwordHash )
```

addUser - adds a new user to the database

Parameters

user	
passwordHash	

4.11.2.5 areContacts()

```
bool SQL::areContacts (
             User * currentUser,
             User * user )
```

areContacts - checks if the users are connected

Parameters

currentUser	
user	

Returns

bool

4.11.2.6 getGroupsByTopic()

```
QVector< Chatroom > SQL::getGroupsByTopic (
            QString topic )
```

getGroupsByTopic - returns the repective user's groups

Parameters

topic	

Returns

QVector<Chatroom>

4.11.2.7 getMembersByTopic()

```
QVector< User > SQL::getMembersByTopic (
            QString topic )
```

getMembersByTopic - returns the members subscribe to the respective topic

Parameters

topic	

Returns

QVector<User>

4.11.2.8 getMessageById()

```
Message * SQL::getMessageById (
            QString id )
```

getMessageById - returns the message information by id

Parameters

id	

Returns

Message

4.11.2.9 getMessagesByTopic()

```
QVector< Message > SQL::getMessagesByTopic (
            QString topic,
            int limit = 30 )
```

getMessagesByTopic - returns the last 30 messages for respective topic

Parameters

topic	
limit	(30 messages)

Returns

QVector<Message>

4.11.2.10 getSubscriptionsByUser()

```
QVector< QString > SQL::getSubscriptionsByUser (
            User * user )
```

getSubscriptionsByUser - returns subscriptions for respective user

Parameters

user	

Returns

QVector<QString>

4.11.2.11 getUserByEmail()

```
User * SQL::getUserByEmail (
            QString email )
```

getUserByEmail - returns the user's information by email

Parameters

email	

Returns

User

4.11.2.12 getUserByUsername()

```
User * SQL::getUserByUsername (
            QString username )
```

getUserByUsername - returns the user's information by username

Parameters

username	

Returns

User

4.11.2.13 getUserContacts()

```
QVector< User > SQL::getUserContacts (
            User * user )
```

getUserContacts - returns vector with the user's contacts

Parameters

user	

Returns

QVector<User>

4.11.2.14 getUserDms()

```
QVector< Chatroom > SQL::getUserDms (
            User * user )
```

getUserDms - returns the direct messages of the respective user

Parameters

user	

Returns

QVector<Chatroom>

4.11.2.15 isGroupAdmin()

```
bool SQL::isGroupAdmin (
            User * user,
            QString topic )
```

isGroupAdmin - returns true if the respective user is the group's admin

Parameters

user	
topic	

Returns

bool

4.11.2.16 isValidLoginByEmail()

```
bool SQL::isValidLoginByEmail (
            QString email,
            QString passwordHash )
```

isValidLoginByEmail - checks if the login by email is valid

Parameters

email	
passwordHash	

Returns

bool

4.11.2.17 isValidLoginByUsername()

```
bool SQL::isValidLoginByUsername (
            QString username,
            QString passwordHash )
```

isValidLoginByUsername - checks if the login by username is valid

Parameters

username	
passwordHash	

Returns

bool

4.11.2.18 removeContact()

```
void SQL::removeContact (
            User * currentUser,
            User * user )
```

removeContact - removes the contact for respective user

Parameters

currentUser	
user	

4.11.2.19 removeGroup()

```
void SQL::removeGroup (
            Chatroom * group )
```

removeGroup - removes group information from database

Parameters

group	

4.11.2.20 removeSubscriptions()

```
void SQL::removeSubscriptions (
            QString topic )
```

removeSubscriptions - removes the stored subscription from the database

Parameters

topic	

4.11.2.21 updateGroup()

```
void SQL::updateGroup (
            Chatroom * group )
```

updateGroup - updates the group information

Parameters

group	

4.11.2.22 updateUser()

```
void SQL::updateUser (
            User * user )
```

updateUser - updates the user information

Parameters

user	

4.11.2.23 validateTopic()

```
QString SQL::validateTopic (
            QString topic )
```

validateTopic - validates the topic name

Parameters

topic	

Returns

QString

4.12 User Class Reference

The User class.

`#include <user.h>`

Public Member Functions

- User (QString username, QString email, QString firstName, QString lastName)

  Constructor for User class.
- int **getId** () const
- QString **username** () const
- void **setUsername** (const QString &username)
- QString **email** () const
- void **setEmail** (const QString &email)
- QString **getFirstName** () const
- void **setFirstName** (const QString &firstName)
- QString **getLastName** () const
- void **setLastName** (const QString &lastName)
- QString **getDisplayName** () const
- void **setDisplayName** (const QString &displayName)

4.12.1 Detailed Description

The User class.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 User()

```
User::User (
            QString username,
            QString email,
            QString firstName,
            QString lastName )  [explicit]
```

Constructor for User class.

Parameters

username	
email	
firstName	
lastName	

Index