Dungeon Stroll

AUTHOR

Версия

Пт 18 Июн 2021

# Оглавление

Table of contents

# Иерархический список классов

## Иерархия классов

Иерархия классов.

QObject

DataBase pagenum

MyTcpServer pagenum

# Алфавитный указатель классов

## Классы

Классы с их кратким описанием.

**DataBase**  pagenum

**MyTcpServer**  pagenum

# Классы

## Класс DataBase

#include <database.h>

Базовые классы:QObject.

### Открытые члены

**DataBase** (QObject \*parent=0)

**~DataBase** ()

*конструктор*

bool **openDataBase** ()

*Внутренние методы для работы с базой данных*

bool **connectToDataBase** ()

*метод открытия базы данных*

void **closeDataBase** ()

*Методы закрытия базы данных*

QString **bdquery** (QString temp)

*метод закрытия базы данных*

### Подробное описание

для коммуникации объектов через сигналы и слоты для распознания языка SQL для запросов к базе данных для данных об ошибках отвечает за соединение с базой данных вывод информации для отладки

### Методы

#### QString DataBase::bdquery (QString *temp*)

метод закрытия базы данных

уточняем, к какой базе данных будет производиться запрос

выполняем запрос, используя введённую в методе переменную

сдвигаем результат запроса с положения "-1"

проверяем, что если длина значения первой строки ответа на запрос больше нуля, то запрос считается выполненным

#### bool DataBase::openDataBase ()

Внутренние методы для работы с базой данных

База данных открывается по заданному пути и имени базы данных, если она существует

qDebug()<<"db is open"; если сработал метод класса QSqlDatabase open(), тогда выводится сообщение, что база данных открыта

qDebug()<<"db not open";

#### Объявления и описания членов классов находятся в файлах:

database.h

database.cpp

## Класс MyTcpServer

#include <mytcpserver.h>

Базовые классы:QObject.

### Открытые слоты

void **slotNewConnection** ()

void **slotClientDisconnected** ()

void **slotServerRead** ()

*освобождение слота, в случае, если клиент отключился*

### Открытые члены

**MyTcpServer** (QObject \*parent=nullptr)

**~MyTcpServer** ()

*конструктор и деструктор предоставляются классом QTcpServer.*

### Подробное описание

предоставляет TCP сервер предоставляет сокет для установления TCP соединения для передачи потоков данных предоставляет некоторые API для приложений, использующих TCP/IP для операций с массивом байтов для процесса отладки

### Конструктор(ы)

#### MyTcpServer::MyTcpServer (QObject \* *parent* = nullptr)[explicit]

конструктор сервера

динамически создаём TCP-сервер

если нет ответа от сервера с любым IP-адресом и портом 33333, то выведется сообщение

о том, что сервер не запущен

обозначает, что сервер работает

вывод в консоль о состоянии сервера

#### MyTcpServer::~MyTcpServer ()

конструктор и деструктор предоставляются классом QTcpServer.

деструктор TCP сервера

определяет статус сервера

закрывает сервер

### Методы

#### void MyTcpServer::slotClientDisconnected ()[slot]

закрываем сокет

выводим в консоль информацию о том, что клиент отключился

#### void MyTcpServer::slotNewConnection ()[slot]

если сервер имеет активный статус

создаётся объект QTcpSocket

вывод сообщения о том, что клиент подключён

#### void MyTcpServer::slotServerRead ()[slot]

освобождение слота, в случае, если клиент отключился

Переменная, для считывания информации с сервера

считываем информацию с клиета

переменная, в которую будем вводить логин

переменная, в которую будем вводить пароль

обозначаем переменную, отвечающую за разделитель

находим логин

ищем разделитель в сообщение с сервера

считываем информацию до разделителя

удаляем информацию о логине в сообщении

находим пароль

ищем разделитель

считываем информацию до разделителя и добавляем её в переменную с паролем

удаляем информацию о пароле

отправляем логин и пароль в функцию авторизации

вывидим в консоль логин

вывидим в консоль пароль

выводим результат авторизации

удаляем информацию из переменной

добавляем в переменную информацию о авторизации

отправляем клиенту информацию об авторизации

#### Объявления и описания членов классов находятся в файлах:

mytcpserver.h

mytcpserver.cpp

# Алфавитный указатель

INDEX