Class name: Database

Header:

#include <QtSql/QSqlDatabase>

#include <QDebug>

#include <QMessageBox>

#include <QApplication>

#include <QSqlQuery>

#include <QSqlError>

#include <stdio.h>

#include <QThread>

#include <nvidianvml.h>

#include "gpumonitor.h"

#include <Windows.h>

#include <QtCharts/QLineSeries>

Inherits: QThread

Name space: std

Description: The Database use the SQLite database system to store data. It will store the GPUs parameters (including temperature, GPU clock, memory clock, fan speed, power draw and GPU id) and overall mining information (including accepted shares, invalid shares, rejected shares and latency) and mining information for each GPU (including GPU id, hash rate, accepted shares, invalid shares and rejected shares). After started, it will run in a new thread to handle the operation of data from class MainWindow.

Constructor:

**Database**():

Parameters: None.

Return: None.

Task: Loading the SQLite driver and initialize the connection with database file (mimerDatabase.db). Initialize all variables and set signal bits to 0.

Destructor:

~***Database***():

Parameters: None.

Return: None.

Task: Deconstructe class and free some allocated variables.

Key Private Variables:

|  |  |
| --- | --- |
| QSqlDatabase \_db; | Connection to database |

Private Types:

None.

Private Methods:

None.

Reimplemented Private Methods:

None.

Private Slots:

None.

Key Public Variables:

|  |  |
| --- | --- |
| int \_insert; | The insert signal bit, 0 means no insert task, 1 means needs to insert data. |
| int \_retrieve; | The retrieve signal bit, 0 means no retrieve task, 1 means needs to retrieve data. |
| int \_insertBusy; | The insert operation signal. 0 means there is no insert operation performing currently while 1 means the thread is inserting data. |
| int \_retrieveBusy; | The retrieve operation signal. 0 means there is no retrieve operation performing currently while 1 means the thread is retrieving data. |
| QList<GPUInfo> \* \_gpusInfoBuffer; | The buffer holds the GPUInfo for inserting. |
| MiningInfo\* \_miningInfoBuffer; | The buffer holds the MingInfo for inserting. |
| QList<QtCharts::QLineSeries \*> \* \_seriesPtr; | The lines in the \_chartHistory. |
| QtCharts::QChart\* \_chartHistory; | The history information chart. |
| QStringList \* searchResultBuffer; | The buffer holds the search result. |
| QStringList \* searchConditionBuffer; | The buffer holds the search conditions to retrieve history information. |

Public Types:

None.

Public Methods:

void **Get\_HistoryNew**(const char\* date1,const char\* date2,int num):

Parameters: search start date, end date and device number.

Return: None.

Task: Retrieve the data from database according with the conditions and plot the line chart of the information.

void **InsertDataNew**():

Parameters: None.

Return: None.

Task: Insert data holds in the buffer to database.

Reimplemented Public Methods:

void ***run***() override:

Parameters: None.

Return: None.

Task: Polling the signal bits and perform the insert from buffer to database and plot the history information according with the conditions sets in the condition buffer.

Logic of Algorithm: 图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

Public Slots:

None.

Signals:

None.

Static Public Members:

None.

Protected Methods:

None.

Static Protected Members:

None.