Project name: MinersCoffeeTest

Main script: tst\_generaltest.cpp

1. The purpose of the test

Testing is the process of executing a program to discover errors in the program. The purpose of testing is to find as many errors in the software as possible before it is put into production operation. Successful test can find the errors in the system and make the system run correctly.

1. Test dependent conditions

The test project and the program project are two opposite Qt project. Rely on QTestLib, the software project needs to rely on this platform and the environment needs to be integrated.

3. Testing Framework:

QTestLib is a unit testing framework provided by Qt for programs or libraries written on Qt. QTestLib provides the basic functionality of the unit testing framework and provides extended functionality for GUI testing.

1. Test initialization:

private:

MainWindow\* w;

Helper helper;

public:

**GeneralTest**();

~***GeneralTest***();

private:

void **initTestCase**();

Parameters: None.

Return: None.

Task: Initialize private variables MainWindow w.

void **cleanupTestCase**();

Parameters: None.

Return: None.

Task: Stop MiningCore and delete MainWindow w.

1. Test preparation:

private:

void **GetTestData**(QList<QString>& input, QList<QString>& result, const QString& in\_filename, const QString& res\_filename);

Parameters: Pointers to input and results’ data file data and filename.

Return: None.

Task: Input the parameter data and expected results in pairs to compare for tests

void **ShowDataError**(const QString& filename1, const QString& filename2);

Parameters: Pointers to input data filename and result data filename.

Return: None.

Task: Warn errors’ location when tests goes wrong.

1. Test hierarchy
2. Unit Tests:
   1. CMD

void **test\_Cmd**():

Task: Check if we get right information of the disk memory size from Wincmd.

Testdata: Directly passed

Testdata size: 1

* 1. GPU Monitoring & Overclocking

void **test\_NvidiaapiSetTempLimit**();

Task: Check if program reset the gpu temprature limit correctly

Testdata: Passed by function **test\_NvidiaapiSetTempLimit\_data**()

Testdata size: 5

void **test\_NvidiaapiSetGPUoffset**();

Task: Check if program reset the gpu offset correctly

Testdata: Passed by function **test\_NvidiaapiSetGPUoffset\_data**()

Testdata size: 5

void **test\_NvidiaapiSetMemoffset**();

Task: Check if program reset the memory offset correctly

Testdata: Passed by function **test\_NvidiaapiSetMemoffset\_data**()

Testdata size: 5

void **test\_NvidiaapiGetTemp**();

Task: Check if program successfully access to the nvidia api and get gpuinfo.

Testdata: Directly checked by function callable answer

Testdata size: 1

void **test\_NvidiaapiComponent**();

Task: Check if program successfully access to the nvidia api and get the temperature.

Testdata: Passed by function **test\_NvidiaapiComponent\_data**()

Testdata size: 5

void **test\_NvidiaapiControlTest**():

Task: Check if program successfully control the gpuinfo

Testdata: Directly passed

Testdata size: 3

* 1. Network Module

void **test\_GetURLInternal**();

Task: Check if program successfully connect to sparkpool

Testdata: Directly passed

Testdata size: 1

* 1. Json Parsing Module

void **test\_ParseJsonForMining**();

Task: Check if program successfully access to the pool api and get mining information.

Testdata: Passed by function **test\_ParseJsonForMining\_data**()

Testdata size: 3

QString input\_filename = "test\_ParseJsonForMining\_input.txt";

QString result\_filename = "test\_ParseJsonForMining\_result.txt";

void **test\_ParseJsonForPool**();

Task: Check if program successfully access to the pool api and get pool information.

Testdata: Passed by function **test\_ParseJsonForPool\_data**()

Testdata size: 3

QString input\_filename = "test\_ParseJsonForPool\_input.txt";

QString result\_filename = "test\_ParseJsonForPool\_result.txt";

* 1. Database

void **test\_Database\_getAdvice**();

Task: Check if program successfully connect to database and get the advice data

Testdata: Directly check

Testdata size: 11

* 1. User Interface

void **test\_ui\_MiningArgsLineEdit**();

void **test\_ui\_MiningArgsLineEdit\_data**();

void **test\_ui\_MiningArgsComboBox**();

void **test\_ui\_MiningArgsComboBox\_data**();

void **test\_ui\_TempPieChart**();

void **test\_ui\_TempPieChart\_data**();

void **test\_ui\_HashrateLineChart**();

void **test\_ui\_HashrateLineChart\_data**();

1. Component/Subsystem Tests:

void **test\_TempPieChart**();

void **test\_HashrateLineChart**();

void **test\_MiningCore**();

void **test\_MiningArgs**();

void **test\_ParsePoolInfo**();

1. Full System Test:

void **test\_FullSystem**()

1. Nonfunctional tests

Security: Thanks to SQLite, we don’t need users’ database password to access to database.

Flexibility: The term refers to the ease with which the application can work with different models of NVIDIA graphics cards.

7. Test Results