

Select an Agent  
to view its workspace

管理知识库

🔥 Knowledge Base Off

Seed 选择模式现在设置为 "Off", 你的知识库将不会被使用。

优化代码, 将Task8生成的随机矩阵显示在UI上, 将TaskE产生的1000个随机数也显示在UI上。

我 12:10 16:05

Reasoning Process >

这是一个在保持 **零报错**、**UI现代化** 的基础上, 专门针对 **Task B (矩阵)** 和 **Task E (随机数)** 进行了 **数据可视化增强** 的版本。

主要修改点:

1. **Task B (矩阵乘法)**:
  - a. 虽然 200x200 的矩阵包含 40,000 个元素 (全部打印会导致界面卡死且无法阅读), 但我实现了\*\*智能预览\*\*。
  - b. 它会在日志中格式化打印出矩阵的 **前 10 行 x 10 列**, 并整齐对齐, 让用户能直观看到“生成的随机矩阵”长什么样, 同时保留后续的计算耗时统计。
2. **Task E (随机数统计)**:
  - a. 生成 1000 个随机数后, 不再只是计算结果, 而是将 **所有 1000 个数字** 以紧凑的矩阵形式 (每行 20 个) 打印到日志窗口中。
  - b. 最后附带均值和方差的统计结果。
3. **UI 细节**:
  - a. 增大了日志缓冲区的限制, 确保能容纳大量文本。
  - b. 优化了数字的格式化 ( ``iomanip`` ), 使输出整齐美观。

完整代码 (main.cpp)

请全选复制代码覆盖原有文件, 直接编译运行。

```
cpp

/*
    项目名称: 高级任务调度器 (数据可视化增强版)

    修改重点:
    1. Task B: 在日志中格式化显示生成的矩阵数据 (预览前10x10区域, 防止4万个数据刷屏卡死UI)。
    2. Task E: 在日志中完整打印 1000 个随机数, 自动换行排版。
    3. 保持所有 UI 优化、零报错修复和 Windows Visual Styles。
*/

// =====
// 1. 链接库与 Manifest 配置
// =====
#pragma comment(Linker, "\\manifestdependency:type='win32' name='Microsoft.Windows.Common-Control")

#pragma comment(Linker, "/SUBSYSTEM:WINDOWS")
#pragma comment(Lib, "user32.lib")
#pragma comment(Lib, "gdi32.lib")
#pragma comment(Lib, "comctl32.lib")

#define _CRT_SECURE_NO_WARNINGS
#define NOMINMAX
#define WIN32_LEAN_AND_MEAN

#include <windows.h>
#include <commctl.h>
#include <string>
#include <vector>
#include <list>
#include <mutex>
#include <thread>
#include <fstream>
#include <memory>
#include <chrono>
#include <ctime>
#include <sstream>
#include <random>
#include <iomanip>
#include <algorithm>

using namespace std;

// =====
// 全局常量与 ID
// =====
#define WM_UPDATE_LOG (WM_USER + 1)
#define WM_UPDATE_LIST (WM_USER + 2)

enum {
    ID_BTN_A = 101, ID_BTN_B, ID_BTN_C, ID_BTN_D, ID_BTN_E,
    ID_BTN_REVOKE,
    ID_EDIT_LOG,
    ID_LIST_TASKS
};

HWND hGlobalWnd = NULL;
HFONT hFontUI = NULL;
HFONT hFontBold = NULL;
HFONT hFontLog = NULL;
HBRUSH hBrushBg = NULL;

// =====
// 日志系统
// =====
class LogWriter {
    ofstream logFile;
    mutex logMutex;
    LogWriter() {
        logFile.open("scheduler.log", ios::app);
    }
public:
    static LogWriter& Instance() { static LogWriter i; return i; }
    ~LogWriter() { if (logFile.is_open()) logFile.close(); }

    void Write(const string& msg) {
        lock_guard<mutex> lock(logMutex);
        auto now = chrono::system_clock::to_time_t(chrono::system_clock::now());
        struct tm t; localtime_s(&t, &now);
        if (logFile.is_open()) {
            logFile << put_time(&t, "[%Y-%m-%d %H:%M:%S] ") << msg << endl;
        }
    }
};

void Log(const string& msg) {
    LogWriter::Instance().Write(msg);
    string* pMsg = new string(msg);
    // 异步发送消息, 防止大文本阻塞工作线程
    if (hGlobalWnd) PostMessageA(hGlobalWnd, WM_UPDATE_LOG, 0, (LPARAM)pMsg);
}

// =====
// 任务系统
// =====
class ITask {
public:
    virtual string GetName() const = 0;
    virtual void Execute() = 0;
    virtual ~ITask() = default;
};

// --- Task A: C:\Data -> D:\Backup ---
class TaskBackup : public ITask {
public:
    string GetName() const override { return "Task A: File Backup (C:\\Data -> D:\\Backup)"; }
    void Execute() override {
        auto now = chrono::system_clock::to_time_t(chrono::system_clock::now());
        struct tm t; localtime_s(&t, &now);
        stringstream ss; ss << "Backup_" << put_time(&t, "%Y%m%d") << ".zip";
        string zipFileName = ss.str();

        string srcPath = "C:\\Data";
        string destDir = "D:\\Backup";
        string destFile = destDir + "\\\" + zipFileName;

        Log("A: Starting backup process...");

        if (!CreateDirectoryA(srcPath.c_str(), NULL) && GetLastError() != ERROR_ALREADY_EXISTS) {
            Log("A: Warning - Check C:\\Data permission.");
        } else {
            ofstream f(srcPath + "\\readme.txt");
            f << "Log: " << time(0) << endl;
            f.close();
        }

        if (!CreateDirectoryA(destDir.c_str(), NULL) && GetLastError() != ERROR_ALREADY_EXISTS) {
            if (GetFileAttributesA("D:\\") == INVALID_FILE_ATTRIBUTES) {
                Log("A: Error - D: drive missing.");
                return;
            }
        }

        Log("A: Compressing to " + destFile + " ...");
        string cmd = "powershell -command \"Compress-Archive -Path '" + src
        int res = (int)WinExec(cmd.c_str(), SW_HIDE);

        if (res > 31) {
            for(chrono::seconds(2));
            and issued.");
        }

        Log("A: Failed to start PowerShell.");
    }
};
```

追问当前话题

