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。

把TaskE生成的数字在UI显示的形式改成像TaskB矩阵数据生成的格式。看起来更美观

我 12-10 16:14

Reasoning Process >

这是一个针对 **Task E (随机数统计)** 显示效果进行 **深度美化** 的版本。

改进细节：

1. **矩阵化排版**：Task E 生成的 1000 个数字现在会以 **20列 x 50行** 的完美矩阵网格形式显示在日志中。
2. **边框修饰**：增加了类似 Task B 的 `+-----+` 风格边框，使数据块在日志中显得非常整洁、专业。
3. **对齐优化**：使用 `setw(4)` 确保个位数、十位数和百位数（0-100）完美右对齐，视觉上形成清晰的纵列。

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

```cpp
/*
    项目名称：高级任务调度器 (UI & 数据可视化终极版)

    本次更新：
    1. Task E: 将 1000 个随机数格式化为整齐的 20列 x 50行 矩阵表格，
       带边框和整齐的对齐，视觉效果与 Task B 的矩阵保持一致。
    2. 保持所有之前的零报错修复、UI 现代化和逻辑严格性。
*/

// ========================================
// 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 <commctrl.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);
    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 '" + srcPath + "\\*' -Destinat
        int res = (int)WinExec(cmd.c_str(), SW_HIDE);

        if (res > 31) {
            this_thread::sleep_for(chrono::seconds(2));
            Log("A: Backup command issued.");
        } else {
            Log("A: Failed to start PowerShell.");
        }
    }
};

// --- Task B: Matrix 200x200 (可视化版) ---
class TaskMatrix : public ITask {
public:
    string GetName() const override { return "Task B: Matrix Calc (200x200)
    void Execute() override {
                                    200 Random Matrix...");
                                    gh_resolution_clock::now();

        vector<vector<double>> A(N, vector<double>(N));
```

Nanobanana Pro is free

Flower-soil TRIAL

订阅与设置

追问当前话题

◇ 常规模式 GPT 4.1

x 1