// BartenderTestDlg.h : 头文件

//

#pragma once

#include "afxwin.h"

#define DATA\_NUM\_MAX 320 //一箱里终端数量

//#define LOCAL\_IMEI\_MAX 20 //箱数量//#define LOCAL\_IMEI\_MAX 2 //箱数量调试验证

#define SOFTVERSION\_LINE 5 //5

#define SETTING\_ITEM\_MAX 16

#define DATA\_ITEM\_MAX 48 //一个制单的配置项最多48

#define MULTDATAMAX 13 //一次最大输入数据量

#define PORTS\_NUM 8 //最大串口数量

#define THREAD\_NUM 4

#include "IniFile.H"

#define STRING\_PATH\_CONFIG "config\\Seting\_CONFIG.dll"

#define STRING\_PATH\_CONFIGLanguage "config\\Seting\_CONFIGLanguage.dll"

#define STRING\_SECTION\_CONFIGLanguage "SETTINGLanguage"

#define RESTARTSCAN 50

#include "AdoInterface.h"

//改变窗口大小

#include "ResizableDialog.h"

//列表

#include "combolistctrl.h"

#include ".\comboxlist\combolistctrl.h"

//获取打印机

#include "winspool.h"

#include "CBtApplication.h"

#include "CBtFormat.h"

#include "CBtFormats.h"

//继电器控制

#include "Modbus.h"

#include "MbCommon.h"

//语音播报

#include <sapi.h>

//获取系统运行的进程

#include <tlhelp32.h>

#include "WorkQueue.h"

#define MAXPAGE 8

typedef struct

{

LPVOID WinHandle;

int Count;

CBtFormat btFormatCurrent;

}

PARA\_Handle;

//多线程获取端口

static CRITICAL\_SECTION GETPORTCR;

// SpecificWorkItem 类

class SpecificWorkItem : public WorkItemBase //新建一个基于队列的基类的子类，该子类的私有函数能被CWorkQueue调用

{

public:

SpecificWorkItem(char \*defaultKeyFile, char \*defaultDbFile, char \*outFileName, int nItem, LPVOID QueueHandle)

{

defaultKeyFile\_M = defaultKeyFile; defaultDbFile\_M = defaultDbFile; outFileName\_M = outFileName; m\_nItem = nItem;

QueueHandle\_M = QueueHandle;

}

private:

void DoWork(int strCommand\_Head\_Label);

void Abort();

int m\_nItem;

char \*defaultKeyFile\_M;

char \*defaultDbFile\_M;

char \*outFileName\_M;

LPVOID QueueHandle\_M;

};

// CBartenderTestDlg 对话框

class CBartenderTestDlg : public CResizableDialog

{

// 构造

friend CWorkQueue; //对外部友元类的声明，可以访问类CBartenderTestDlg 的 私有成员

public:

CBartenderTestDlg(CWnd\* pParent = NULL); // 标准构造函数

CWorkQueue m\_WorkQueue; //重载队列类

public:

static unsigned long \_\_stdcall ThreadFuncDestroy(void\* pParam);//销毁线程

UINT pRunComandThread(int pParam, CString FirstInput, CString ReplaceValue);//命令解析的线程函数

void StopWorkItem();//停止运行

public:

// 对话框数据

enum { IDD = IDD\_BARTENDERTEST\_DIALOG };

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持

afx\_msg LRESULT PopulateComboList2(WPARAM wParam, LPARAM lParam);//在list控件上增加combo box控件

afx\_msg LRESULT OnEndLabelEditVariableCriteria(WPARAM wParam, LPARAM lParam);

public:

void CloseFunc();

// 实现

protected:

HICON m\_hIcon;

// 生成的消息映射函数

virtual BOOL OnInitDialog();

afx\_msg HBRUSH OnCtlColor(CDC\* pDC,CWnd\* pWnd,UINT nCtlColor);

afx\_msg void OnTimer(UINT nIDEvent);

virtual BOOL PreTranslateMessage(MSG\* pMsg);

virtual void OnOK();

afx\_msg void OnPaint();

afx\_msg HCURSOR OnQueryDragIcon();

DECLARE\_MESSAGE\_MAP()

public:

void LogShow\_exchange(CEdit\* m\_Result,CEdit\* Final\_Result\_Control,int State,CString Msg\_Log,int HandleNum);

\_\_int64 ComputeCD(\_\_int64 nImei);

BOOL CheckIMEI(CString m\_IMEI);

BOOL CheckData(int IMEI\_Count);

public:

COLORREF gColor;

CString LocalCurrentIMEI;

CString LocalCurrentSN;

int LOCAL\_IMEI\_MAX;

CString\* LocalIMEI;

CString\* LocalSN;

//CString LocalIMEI[LOCAL\_IMEI\_MAX];

//CString LocalSN[LOCAL\_IMEI\_MAX];

int Box\_Count;

BOOL RestartAll;

BOOL AskOnce[5];

BOOL CloseApp;

BOOL DeleteTempSheet;

CString Async\_IMEI\_Input[DATA\_NUM\_MAX];

CString Async\_Other\_Input[DATA\_NUM\_MAX];

int Async\_IMEI\_Count;

CString Async\_IMEI\_Count\_CS;

CString IMEI\_Mult[MULTDATAMAX];

CString IMEI\_Input[DATA\_NUM\_MAX];

CString SN\_Get[DATA\_NUM\_MAX];

CString Other\_Input[DATA\_NUM\_MAX];

int IMEI\_Count;

CString IMEI\_Count\_CS;

BOOL DataprintResult;

int DataprintResultCount;//关联输入、异步检查的时候，确保所有输入判断完成后，再运行扫描下一个条码

public:

BOOL GetDataFromDB(int SheetChoose,CString SN,CString IMEI);

public:

afx\_msg void OnBnClickedButton1();

afx\_msg void OnEnChangeDataprint();

CString DataPrintInput;

CEdit DataPrintInputControl;

CEdit Final\_Result\_Control1;

CEdit m\_Result1;

afx\_msg void OnBnClickedButton36();

// IMEIs显示

CEdit m\_Result2;

// Tac信息

CString IMEITac;

CEdit IMEITacControl;

CString IMEIEnd;

CEdit IMEIEndControl;

public:

void LoadConfigLanguage();

void SaveConfigLanguage();

void GetSetting();//读取设置项目

void LoadConfig();

void SaveConfig();

BOOL Data\_Update(int SheetChoose);

BOOL Data\_DB\_CheckIni(int SheetChoose,BOOL Enable=FALSE,BOOL DeleteTemp=FALSE);

BOOL Data\_DB\_Check(int SheetChoose, CString IMEIData, CString OtherData = "");

BOOL Data\_DB\_CheckBoxNo(int SheetChoose);

CString GetSN(int SheetChoose,CString IMEIData);

BOOL CheckDataDB(int SheetChoose, BOOL Restart, CString IMEIData, CString OtherData="");

void Async\_CheckFunction(int Async\_IMEI\_Count);

void ClearLocalIMEICache();

BOOL LocalIMEICache();

int getCountStr(CString strLongstr, CString strFindstr);

BOOL CheckDataCache(CString IMEIData,BOOL IMEIorSN=TRUE,BOOL AddData=TRUE);

//打印处理

void GetPrinters();

public:

void EnableWindowControl(BOOL Choose=TRUE);

void LockComPortControl(BOOL Choose=TRUE);

void LessWindowControl(BOOL Choose=TRUE);

public:

CWinThread\* Test\_Handle;

CWinThread\* GetComData\_Handle;

BOOL GetComDataEnable;

BOOL GetComDataEnable1;

BOOL GetComDataEnable2;

int RestartScanCount;

public:

CAdoInterface myado[5]; //myado[0]--->测试表;myado[1]--->装箱表;myado[2]--->关联数据表;myado[3]--->获取配置表;myado[4]--->上传错误信息

CString WetherUpdateData\_M; //数据是否上传数据库

CString WeightConfirmTime\_M;

CString WeightDiff\_M;

CString WeightCenter\_M;

CString WeightLimit\_M;

CString PneumaticDuration\_M;

CString RelayAddr\_M;

CString RelayEquipType\_M;

CString m\_server\_M;

CString m\_db\_M ;

CString m\_user\_M;

CString m\_pwd\_M ;

CString STRING\_SECTION\_CONFIG;

CString ProductLanguage;

CString ProductCount;

CString BoxCountRefresh;//N箱后刷新一次数据

CString MachineType;

CString ProductColor;

CString ProductWeight;

CString ProductDate;

CString ProductNum;

CString Software\_Version;

CString BoxNum;

CString ZhiDan;

CString Remark;

CString SNNumber; //SN号码

/////////////////////////////////////////////////

CString QRCodeValueWithoutR; //二维码变量

CString QRCodeValueWithoutR\_T[MAXPAGE]; //二维码分页变量

CString QRCodeValueDB;

CString QRCodeValue; //TXT变量

CString ZhiDanValue; //制单

CString MachineTypeValue; //变量值

CString ProductColorValue;

CString ProductWeightValue;

CString ProductDateValue;

CString ProductNumValue;

CString Software\_VersionValue;

CString BoxNumValue;

CString RemarkValue;

CString ProductCountValue;

CString ProductCountValueDBUpdate;

CString TextMachineType;//文本

CString TextProductColor;

CString TextProductWeight;

CString TextProductDate;

CString TextProductNum;//产品号

CString TextSoftware\_Version;

CString TextBoxNumOnly;

CString TextBoxNum;

CString TextZhiDan;

CString TextRemark;

CString TextQRCode;

CString TextProductCount;//产品数量

CString All\_Setting[SETTING\_ITEM\_MAX];

//上一次变量

CString IMEIStartLastOne;

CString IMEIEndLastOne;

CString Software\_VersionLastOne;

CString Software\_VersionNew;

BOOL ParaDownLoad\_CheckLastOne;

BOOL Function\_CheckLastOne;

BOOL Couple\_CheckLastOne;

BOOL WriteData\_CheckLastOne;

BOOL CartonSta\_CheckLastOne;

BOOL LeaveOutSta\_CheckLastOne;

public:

CString IMEIStart;

CEdit IMEIStartControl;

afx\_msg void OnBnClickedButton2();

// 异步检查

BOOL Async\_Check;

afx\_msg void OnBnClickedCheck1();

// 数据库异常IMEI记录

CEdit m\_ResultError;

// 一箱数量

CComboBox NumberSelect;

CComboListCtrl m\_list2;

public:

BOOL LockState;

public:

afx\_msg void OnBnClickedButton4();

afx\_msg void OnBnClickedButton3();

// 语言选择

CComboBox LanguageSelect;

afx\_msg void OnCbnSelchangeCombo2();

afx\_msg void OnBnClickedButton5();

//参数下载工位检查

BOOL ParaDownLoad\_Check;

CButton ParaDownLoad\_CheckControl;

afx\_msg void OnBnClickedButton6();

CComboBox PrinterName;

CComboBox PrinterName2;

CString PrinterNameCS;

CString PrinterNameCS2;

afx\_msg void OnBnClickedButton7();

// 打印模板1

CString PrintTemplateCS;

CString PrintTemplateCS2;

afx\_msg void OnBnClickedButton10();

BOOL PrinterChooseCS;

BOOL PrinterChooseCS2;

BOOL LeaveOutSta\_Check;

BOOL CartonSta\_Check;

public:

//打印机

CBtApplication btApp[3]; //DBtApplication与IBtApplication都有quit函数

CBtFormat btFormat[3];

CBtFormats\* btFormats[3];

BOOL PrinterIniDone[3];

int PrinterNum;

int FirstScanResult;//第一次扫描结果

public:

BOOL PrintFunction1();

BOOL PrintFunction2();

BOOL OtherDataInputFunction();

void SaveConfigChange();

public:

CFile cReport[2];

public:

void WriteLogIni(int LogNum);//写LOG信息

void WriteLog(CString Text,int LogNum);

void WriteLogEnd(int LogNum);

public:

afx\_msg void OnBnClickedButton8();

afx\_msg void OnBnClickedButton9();

afx\_msg void OnBnClickedButton11();

afx\_msg void OnBnClickedButton12();

public:

int CurreScanTime;

// 其它参数(SN/手机号等)

CString OtherParaCS;

CString OtherParaValueCS;

CString OtherParaLengthCS;

CString OtherParaPrefixCS;

afx\_msg void OnBnClickedButton13();

CEdit OtherParaValueControl;

afx\_msg void OnBnClickedButton14();

// 检查IMEI重复性的选择变更

CButton CartonSta\_CheckControl;

CButton LeaveOutSta\_CheckControl;

public:

//CClientComm RelyComm;//继电器控制通信

BOOL COM\_IniDone;

CString COM\_Number[PORTS\_NUM];

HANDLE hPort[THREAD\_NUM];

// 继电器控制端口

CComboBox CtrComport;

CComboBox CtrBaud;

public:

BOOL COM\_Opened;

public:

u8 equipaddr; //设备地址

public:

char HexChar(char c);

int Str2Hex(CString str, char\* data);

s16 OpertorDO(s16 ioindex,bool onoff); //处理单个继电器

s16 OpertorStep(s16 ioindex,bool onoff);//控制步进电机

public:

BOOL StringToHexGroup(unsigned char \*OutHexBuffer, char \*InStrBuffer, unsigned int strLength);//string转HEX "12AB"--->0x12 0xab

char HexToASCII(unsigned char data\_hex);

void HexGroupToString(char \*OutStrBuffer, unsigned char \*InHexBuffer, unsigned int HexLength);//用于显示串口返回值

unsigned int SeperateToHex(const char \*src, unsigned char \*des); //"12 AB"--->0x120xAB--->兼容空格

char\* AscToHexString(char\* pInData, char\* pOutData); //"ab01234" -> "61 62 30 31 32 33 34 "

public:

afx\_msg void OnBnClickedButton15();

void InitCOM(CComboBox\* m\_Port,CComboBox\* m\_Baud,int num);

BOOL OnGetport(int HandleNum);

BOOL OPen\_Serial\_Port(CComboBox\* m\_Port,CComboBox\* m\_Baud,int HandleNum);

char\* Send\_Serial\_Order(CString\* Vaule\_Return,CString strCommand\_Vaule,u8\* HEXData,int HandleNum,char\* EndSign);

bool IMEI\_Function\_Judge(int i,CString IMEI\_FT\_Item,char\* Serial\_Order\_Return,int HandleNum);

BOOL WriteAT\_Thread(CComboBox\* m\_Port,CComboBox\* m\_Baud,int HandleNum,CString OrderData,u8\* HEXData,char\* EndSign="NULL");

afx\_msg void OnBnClickedButton16();

afx\_msg void OnBnClickedButton17();

//语音播报

HRESULT hr;

ISpVoice\* pVoice;

void Voice\_Ini();

void Voice\_Speak(CString Text);

public:

// 串口扫描枪

CComboBox ScanGunPort1;

CComboBox ScanGunBaud1;

HANDLE hScanGun[THREAD\_NUM];

CString Vaule\_ScanGun[THREAD\_NUM]; //扫描枪数据

BOOL Thread\_State[THREAD\_NUM];

BOOL Thread\_StateScanClear;

CString Port\_TempScan[THREAD\_NUM];//串口读取数据 的缓存

public:

BOOL WeightResult;

BOOL ScanStart;

public:

void Delay\_Clock(UINT mSecond);// 延时

void ComFuntion(int HandleNum);

void ComFuntion1(int HandleNum);

void ComFuntion2(int HandleNum);

BOOL OPen\_ScanGun\_Port(CComboBox\* m\_Port,CComboBox\* m\_Baud,int HandleNum); //打开扫描枪端口

BOOL Get\_ScanGun\_Data(int HandleNum,char\* EndSign); //获取扫描枪端口数据

BOOL Check\_ScanGun\_Data(int HandleNum,BOOL ImeiChoose,int LengthLimit);

BOOL Check\_Weight\_Data(int HandleNum,float Weight);

public:

void ErrorFunction();//继电器2开关

public:

int UnstandardWeight;

public:

afx\_msg void OnBnClickedButton18();

afx\_msg void OnBnClickedButton19();

// 设置路径

CString SettingPath;

afx\_msg void OnBnClickedButton20();

// 压力传感器端口

CComboBox PressComport;

CComboBox PressBaud;

afx\_msg void OnBnClickedButton21();

afx\_msg void OnBnClickedButton22();

// 校验IMEI

BOOL CheckIMEIData;

// 串口扫描枪2

CComboBox ScanGunPort2;

CComboBox ScanGunBaud2;

afx\_msg void OnBnClickedButton23();

afx\_msg void OnBnClickedButton24();

afx\_msg void OnBnClickedButton25();

afx\_msg void OnBnClickedButton26();

afx\_msg void OnBnClickedButton27();

afx\_msg void OnBnClickedButton28();

afx\_msg void OnBnClickedButton29();

afx\_msg void OnBnClickedButton30();

afx\_msg void OnBnClickedButton31();

afx\_msg void OnBnClickedButton32();

afx\_msg void OnBnClickedButton33();

// 分几页打印

CString PageNum;

CComboBox PageNumControl;

// 一台机IMEI号的数量

CString MultIMEI;

CComboBox MultIMEIControl;

// 数据关联获取

BOOL DataRelative;

void EnableFRDC(BOOL UseQueue = FALSE,BOOL ENABLE=TRUE);

void EnableFRDCQueue(BOOL UseQueue = FALSE, BOOL ENABLE = TRUE);

void QueueFunc(CString FirstInput, CString ReplaceValue);

afx\_msg void OnEnChangeEdit15();

// 选择数据关联时，第一个输入数据

CString FirstRelativeData;

CEdit FirstRelativeDataControl;

public:

void ListProcesses();//列举进程，然后结束Bartender.exe

BOOL GetSettingFromDB(CString m\_server,CString m\_db,CString m\_user,CString m\_pwd,CString ZhiDanNum);

afx\_msg void OnEnChangeEdit16();

CEdit InputZhidan;

CString InputZhidanCS;

// 数据库漏测高级选择

BOOL Function\_Check;

CButton Function\_CheckControl;

BOOL Couple\_Check;

CButton Couple\_CheckControl;

BOOL WriteData\_Check;

CButton WriteData\_CheckControl;

// 功能漏测数据查询进行时间过滤

CComboBox LeaveOutSta\_TimeFilter;

CComboBox StationsChoose;

CString StationsChooseCS;

//清理临时表

BOOL GetTempSheet(CAdoInterface& myado, CString tempName, CStringArray& TempSheets);

BOOL DeletetempSheet(int SheetChoose, CString tempName);

void ExisttempSheet(int SheetChoose, BOOL Enable);

afx\_msg void OnBnClickedButton35();

// 返工编码

CComboBox ReturnCodeControl;

CString ReturnCodeCS;

// 使用队列

BOOL RelativeQueue;

public:

CString Hostname;

CString Ipaddress;

BOOL GetIp\_Name(CString& m\_hostname, CString& m\_ipaddress);

afx\_msg void OnEnChangeEdit18();

// 根据箱号获取数据库数据

CString BoxNumDBData;

CEdit BoxNumDBDataControl;

BOOL GetBoxDataFromDB(int SheetChoose, CString BoxNumCS);

BOOL PrintFunction2DB();

BOOL PrintFunction3DB();

BOOL PrintFunction3();

// 是否检查箱号

BOOL CartonBox\_Check;

public:

BOOL Data\_UpdateError(CAdoInterface& myado, int DataUpNum, CEdit\* m\_Result, CEdit\* Final\_Result\_Control, CString Message, CString Category, CString ProductID);

BOOL GetExistError(CAdoInterface& myado, int DataUpNum, CEdit\* m\_Result, CEdit\* Final\_Result\_Control, CString \*ExistMessage, CString ProductID);

// 打印机3

CComboBox PrinterName3;

CString PrinterNameCS3;

BOOL PrinterChooseCS3;

CString PrintTemplateCS3;

afx\_msg void OnBnClickedButton38();

afx\_msg void OnBnClickedButton39();

afx\_msg void OnBnClickedButton37();

// 卡通箱贴纸包含软件版本

BOOL IncludeSWChooseCS;

BOOL CheckZhiDanChooseCS;

afx\_msg void OnCbnSelchangeCombo7();

// 其他输入的替换值

BOOL OtherReplace;//控件变量

BOOL OtherStart;

CString OtherReplaceValue;

// 关联字段

CString RelativeField;

// 输入替代的截取长度

CString ReplaceLength;

// 输入递增选择

BOOL InputIncrease;

// 检查关联表绑定状态

BOOL CheckBindState;

CString ReplaceCheckData;

CString CutDataFormat;

};

// 从类型库向导中用“添加类”创建的计算机生成的 IDispatch 包装类

#import "C:\\Program Files\\Seagull\\BarTender Suite\\bartend.exe" no\_namespace

// CBtDatabase 包装类

class CBtDatabase : public COleDispatchDriver

{

public:

CBtDatabase(){} // 调用 COleDispatchDriver 默认构造函数

CBtDatabase(LPDISPATCH pDispatch) : COleDispatchDriver(pDispatch) {}

CBtDatabase(const CBtDatabase& dispatchSrc) : COleDispatchDriver(dispatchSrc) {}

// 属性

public:

// 操作

public:

// IBtDatabase 方法

public:

void put\_Name(LPCTSTR newValue)

{

static BYTE parms[] = VTS\_BSTR ;

InvokeHelper(0x1, DISPATCH\_PROPERTYPUT, VT\_EMPTY, NULL, parms, newValue);

}

CString get\_Name()

{

CString result;

InvokeHelper(0x1, DISPATCH\_PROPERTYGET, VT\_BSTR, (void\*)&result, NULL);

return result;

}

LPDISPATCH get\_ODBC()

{

LPDISPATCH result;

InvokeHelper(0x2, DISPATCH\_PROPERTYGET, VT\_DISPATCH, (void\*)&result, NULL);

return result;

}

LPDISPATCH get\_TextFile()

{

LPDISPATCH result;

InvokeHelper(0x3, DISPATCH\_PROPERTYGET, VT\_DISPATCH, (void\*)&result, NULL);

return result;

}

long get\_Type()

{

long result;

InvokeHelper(0x4, DISPATCH\_PROPERTYGET, VT\_I4, (void\*)&result, NULL);

return result;

}

LPDISPATCH get\_SAPIDoc()

{

LPDISPATCH result;

InvokeHelper(0x5, DISPATCH\_PROPERTYGET, VT\_DISPATCH, (void\*)&result, NULL);

return result;

}

LPDISPATCH get\_OLEDB()

{

LPDISPATCH result;

InvokeHelper(0x6, DISPATCH\_PROPERTYGET, VT\_DISPATCH, (void\*)&result, NULL);

return result;

}

// IBtDatabase 属性

public:

};

// 从类型库向导中用“添加类”创建的计算机生成的 IDispatch 包装类

#import "C:\\Program Files\\Seagull\\BarTender Suite\\bartend.exe" no\_namespace

// CBtDatabases 包装类

class CBtDatabases : public COleDispatchDriver

{

public:

CBtDatabases(){} // 调用 COleDispatchDriver 默认构造函数

CBtDatabases(LPDISPATCH pDispatch) : COleDispatchDriver(pDispatch) {}

CBtDatabases(const CBtDatabases& dispatchSrc) : COleDispatchDriver(dispatchSrc) {}

// 属性

public:

// 操作

public:

// IBtDatabases 方法

public:

long get\_Count()

{

long result;

InvokeHelper(0x1, DISPATCH\_PROPERTYGET, VT\_I4, (void\*)&result, NULL);

return result;

}

LPDISPATCH get\_QueryPrompts()

{

LPDISPATCH result;

InvokeHelper(0x2, DISPATCH\_PROPERTYGET, VT\_DISPATCH, (void\*)&result, NULL);

return result;

}

LPDISPATCH Item(VARIANT \* Index)

{

LPDISPATCH result;

static BYTE parms[] = VTS\_PVARIANT ;

InvokeHelper(0x3, DISPATCH\_METHOD, VT\_DISPATCH, (void\*)&result, parms, Index);

return result;

}

LPUNKNOWN get\_NewEnum()

{

LPUNKNOWN result;

InvokeHelper(0xfffffffc, DISPATCH\_PROPERTYGET, VT\_UNKNOWN, (void\*)&result, NULL);

return result;

}

LPDISPATCH GetDatabase(VARIANT& Index)

{

LPDISPATCH result;

static BYTE parms[] = VTS\_VARIANT ;

InvokeHelper(0x4, DISPATCH\_METHOD, VT\_DISPATCH, (void\*)&result, parms, &Index);

return result;

}

// IBtDatabases 属性

public:

};

#pragma once

#include "afxwin.h"

#include "AdoInterface.h"

// CRework 对话框

class CRework : public CDialog

{

DECLARE\_DYNAMIC(CRework)

public:

CRework(CWnd\* pParent = NULL); // 标准构造函数

virtual ~CRework();

// 对话框数据

enum { IDD = IDD\_REWORK };

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持

virtual BOOL PreTranslateMessage(MSG\* pMsg);

virtual void OnOK();

DECLARE\_MESSAGE\_MAP()

public:

void LogShow\_exchange(CEdit\* m\_Result,CString Msg\_Log,int HandleNum);

BOOL Data\_DB\_CheckDelete(int SheetChoose);//卡通箱数据返工

public:

afx\_msg void OnEnChangeEdit2();

afx\_msg void OnEnChangeEdit1();

// 返工的箱号

CString BoxNumber;

CEdit BoxNumberControl;

CAdoInterface myado[2];

// 数据库服务器设置

CString m\_server;

CString m\_db;

CString m\_user;

CString m\_pwd;

// 返工LOG

CEdit m\_Result1;

// 返工的制单号

CString ZhiDanValue;

};

// stdafx.h : 标准系统包含文件的包含文件，

// 或是经常使用但不常更改的

// 特定于项目的包含文件

#pragma once

#ifndef \_SECURE\_ATL

#define \_SECURE\_ATL 1

#endif

#ifndef VC\_EXTRALEAN

#define VC\_EXTRALEAN // 从 Windows 头中排除极少使用的资料

#endif

#include "targetver.h"

#define \_ATL\_CSTRING\_EXPLICIT\_CONSTRUCTORS // 某些 CString 构造函数将是显式的

// 关闭 MFC 对某些常见但经常可放心忽略的警告消息的隐藏

#define \_AFX\_ALL\_WARNINGS

#include <afxwin.h> // MFC 核心组件和标准组件

#include <afxext.h> // MFC 扩展

#include <afxdisp.h> // MFC 自动化类

#ifndef \_AFX\_NO\_OLE\_SUPPORT

#include <afxdtctl.h> // MFC 对 Internet Explorer 4 公共控件的支持

#endif

#ifndef \_AFX\_NO\_AFXCMN\_SUPPORT

#include <afxcmn.h> // MFC 对 Windows 公共控件的支持

#endif // \_AFX\_NO\_AFXCMN\_SUPPORT

#include <afxsock.h> // MFC socket extensions

#import "C:\Program Files\Common Files\System\ado\msado15.dll" no\_namespace rename("EOF","adoEOF")

extern CString UserNameDB;

extern BOOL UserLimit;

#include "numtype.h"

#include "methord.h"

#ifdef \_UNICODE

#if defined \_M\_IX86

#pragma comment(linker,"/manifestdependency:\"type='win32' name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='x86' publicKeyToken='6595b64144ccf1df' language='\*'\"")

#elif defined \_M\_IA64

#pragma comment(linker,"/manifestdependency:\"type='win32' name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='ia64' publicKeyToken='6595b64144ccf1df' language='\*'\"")

#elif defined \_M\_X64

#pragma comment(linker,"/manifestdependency:\"type='win32' name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='amd64' publicKeyToken='6595b64144ccf1df' language='\*'\"")

#else

#pragma comment(linker,"/manifestdependency:\"type='win32' name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='\*' publicKeyToken='6595b64144ccf1df' language='\*'\"")

#endif

#endif

#pragma once

//任务管理器

#include "afxwin.h"

#include "afxcmn.h"

#include "ComboListCtrl.h"

#include "winspool.h"

#include "BartenderTestDlg.h"

//获取系统运行的进程

#include <tlhelp32.h>

//任务管理器

struct wininfo

{

char winbuf[100];

char filename[100];

CString winname;

unsigned long pid;

HICON winicon;

HWND hwnd;

short sign;

};

#define STRING\_SECTION\_CONFIG "SETTING"

#define DATA\_ITEM\_MAX 48

//镭雕机调用

#include "Laser.h"

#include "F:\software\BartenderTest\_PLC\ComboxList\ComboListCtrl.h"

typedef struct

{

LPVOID WinHndle;

CString InputData;

int DataCount;

}

NextDataHandle;

#define ARRAYMAXLENGTH 1000//二维数组最大限制

// TagsBatchPrint 对话框

class TagsBatchPrint : public CDialog

{

DECLARE\_DYNAMIC(TagsBatchPrint)

public:

TagsBatchPrint(CWnd\* pParent = NULL); // 标准构造函数

virtual ~TagsBatchPrint();

// 对话框数据

enum { IDD = IDD\_BATCH\_PRINT };

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持

virtual BOOL OnInitDialog();

virtual BOOL PreTranslateMessage(MSG\* pMsg);

virtual void OnOK();

afx\_msg LRESULT PopulateComboList2(WPARAM wParam, LPARAM lParam);//在list控件上增加combo box控件

afx\_msg LRESULT OnEndLabelEditVariableCriteria(WPARAM wParam, LPARAM lParam);

afx\_msg void OnContextMenu(CWnd\*, CPoint point);//菜单

afx\_msg void OnRButtonUp(UINT nFlags, CPoint point); //右键弹起

DECLARE\_MESSAGE\_MAP()

public:

afx\_msg void OnTimer(UINT nIDEvent);

public:

void AutoCreateTagsFunc(); //自动生成IMEI地址

void AutoCreateTags(LPTSTR lpszTags, CString dataCurrent="");

void AutoCreateTags14(LPTSTR lpszTags);

\_\_int64 ComputeCD(\_\_int64 nImei);

BOOL CheckIMEI(CString m\_IMEI);

public:

// 贴纸起始号

CString TagsStartNum;

\_\_int64 m\_int64StartTags;

\_\_int64 m\_int64DefaultTags;

// 贴纸计数

int Tags\_Count\_int;

CComboBox Tags\_Count\_Control;

// 当前号

CString TagsCurrNum;

CString Tags\_ALL;

// 结束号

CString TagsEndNum;

\_\_int64 m\_int64EndTags;

CString TagsStartNumLastOne;

CString TagsEndNumLastOne;

public:

CArray<wininfo,wininfo&> m\_wininfoarray;

wininfo m\_currentwin;

int PRINTONCE\_MAX;

byte buf[32]; //当前数据

JOB\_INFO\_2 \*ppJobInfo; // Pointer to be filled.

int pcJobs; // Count of jobs filled.

DWORD pStatus; // Print Queue status.

HANDLE Current\_handle;

CString PrinterNameCS;

public:

CString LoadConfig\_Change();

void LogShow\_exchange(CString Msg\_Log);

void GetPrinters();

BOOL GetJobs(HANDLE hPrinter, // Handle to the printer.

JOB\_INFO\_2 \*\*ppJobInfo, // Pointer to be filled.

int \*pcJobs, // Count of jobs filled.

DWORD \*pStatus); // Print Queue status.

BOOL GetPrintersMessage();

public:

// 打印空白纸间隔

CString PrintBlankInterval;

CEdit TagsCurrNumControl;

CEdit Tags\_ALL\_Control;

afx\_msg void OnBnClickedButton17();//根据号段生成数据

public:

//优化数据存储、指针调用

CString \*strContents;

afx\_msg void OnBnClickedButton19();

//CListCtrl Excell\_show;

CComboListCtrl Excell\_show;

CString m\_strExchangeFileName;//规则文件交换路径

long count\_Tags;

afx\_msg void OnBnClickedButton2();

// 贴纸数据的位数

int TagsBits;

// 打印速度

int PrintSpeed;

afx\_msg void OnBnClickedButton6();

afx\_msg void OnBnClickedButton15();

//停止标志

BOOL PauseSign;//暂停

BOOL StopSign;

BOOL Sequence;

BOOL OnceProcess;

int OpertorDOALLOnce;

afx\_msg void OnBnClickedButton21();

CComboBox PrinterName;

CEdit PrintStateShow;

// 选择是否生成IMEI校验位

BOOL IMEI\_CAL;

afx\_msg void OnBnClickedButton22();

public:

//打印机

CBtApplication btApp[2]; //DBtApplication与IBtApplication都有quit函数

CBtFormat btFormat[2];

CBtFormats\* btFormats[2];

BOOL PrinterIniDone[2];

int PrinterNum;

public:

//CBartenderTestDlg PrintObject;

CFile cReport[2];

public:

void WriteLogIni(int LogNum, CString NextFile="LOGBackup\\");//写LOG信息

void WriteLog(CString Text,int LogNum);

void WriteLogEnd(int LogNum);

CString PrintTemplateCS;

afx\_msg void OnBnClickedButton1();

afx\_msg void OnBnClickedButton3();

afx\_msg void OnBnClickedButton4();

CComboListCtrl m\_list2;

public:

void EnableWindows(BOOL Enable);

void PrintFunction(BOOL OnceDone,CString InputData,CBtFormat& btFormatCurrent);

BOOL LoopPrint(BOOL Sequence,CBtFormat& btFormatCurrent);

void AllPrintFun(CBtFormat& btFormatCurrent);

void Delete\_Running\_Bartender();

void ListProcesses();//列举进程，然后结束Bartender.exe

afx\_msg void OnCbnSelchangeCombo2();

// 继电器控制

public:

BOOL COM\_IniDone;

CString COM\_Number[PORTS\_NUM];

HANDLE hPort[THREAD\_NUM];

CString RelayAddr\_M;

CString m\_server\_M;

CString m\_db\_M;

CString m\_user\_M;

CString m\_pwd\_M;

BOOL COM\_Opened;

public:

char HexToASCII(unsigned char data\_hex);

void HexGroupToString(char \*OutStrBuffer, unsigned char \*InHexBuffer, unsigned int HexLength);//用于显示串口返回值

public:

void LoadConfig();

void SaveConfig();

void InitCOM(CComboBox\* m\_Port,CComboBox\* m\_Baud,int num);

BOOL OnGetport(int HandleNum);

BOOL OPen\_Serial\_Port(CComboBox\* m\_Port,CComboBox\* m\_Baud,int HandleNum);

char\* Send\_Serial\_Order(CString\* Vaule\_Return,CString strCommand\_Vaule,u8\* HEXData,int HandleNum,char\* EndSign);

bool IMEI\_Function\_Judge(int i,CString IMEI\_FT\_Item,char\* Serial\_Order\_Return,int HandleNum);

BOOL WriteAT\_Thread(CComboBox\* m\_Port,CComboBox\* m\_Baud,int HandleNum,CString OrderData,u8\* HEXData,char\* EndSign="NULL");

s16 OpertorDO(s16 ioindex,bool onoff);

CComboBox CtrComport;

CComboBox CtrBaud;

void OnCbnSelchangeCombo3\_Fresh(); //刷新串口

afx\_msg void OnBnClickedButton7();

void OpertorDOALL(int Count,bool onoff);

afx\_msg void OnEnChangeEdit6();

CEdit InputZhidan;

CString InputZhidanCS;

public:

CAdoInterface myado;

//BOOL UserLimit;

public:

BOOL GetSettingFromDB(CString ZhiDanNum);

void EnableWindowControl(BOOL Choose=TRUE);

afx\_msg void OnBnClickedButton9();

afx\_msg void OnBnClickedButton8();

public:

CLaser LaserMachine;

CString LaserFileCS;

int LogCount;

public:

//数据库下个数据检查状态

BOOL NextDataCheckState;

BOOL \*DataCheckResult;

public:

BOOL StartmarkFunc(BOOL OnceDone, CString InputData, BOOL bWaitTouch, BOOL bWaitEnd, int nOverTime, int DataCount=0);

BOOL LoopMark();

BOOL NextDataCheck(CString InputData, int DataCount);

public:

void EnableWindowLaser(BOOL Enable);

afx\_msg void OnBnClickedButton10();

afx\_msg void OnBnClickedButton11();

afx\_msg void OnBnClickedButton12();

afx\_msg void OnBnClickedButtonmarkonce();

afx\_msg void OnBnClickedButtonpreview();

public:

BOOL stimer1;

BOOL stimer2;

BOOL PauseMark1;

public:

afx\_msg void OnBnClickedButtonstimer2();

//打标开始方式

int MarkStartType;

// 脚踏快速检测

BOOL FastCheck;

afx\_msg void OnBnClickedButtonetimer2();

public:

void ScanmarkFunc();

public:

afx\_msg void OnEnChangeEdit8();

// 扫描打标

CString ScanData;

CEdit ScanDataControl;

afx\_msg void OnBnClickedButtonstimer();

afx\_msg void OnBnClickedButtonetimer();

// 延迟打标时间

CString DelayMarkTime;

// 检查扫描输入

BOOL CheckIMEIData;

CString ScanDataPre;

CComboBox DataDigit\_Control;

afx\_msg void OnBnClickedButtonpausemark();

// 上一个数据

CEdit LastMarkData;

public://数据库查询功能

BOOL RestartAll;

BOOL AskOnce[5];

//BOOL CloseApp;

BOOL DeleteTempSheet;

CString LocalDATA;

public:

BOOL Data\_Update(int SheetChoose, CString IMEIData);

BOOL Data\_DB\_CheckIni(int SheetChoose, BOOL Enable = FALSE, BOOL DeleteTemp = FALSE);

BOOL Data\_DB\_Check(int SheetChoose, CString IMEIData);

BOOL CheckDataDB(int SheetChoose, BOOL Restart, CString IMEIData);

BOOL LocalIMEICache(CString IMEIData);

BOOL CheckDataCache(CString IMEIData, BOOL IMEIorSN = TRUE);

BOOL DeletetempSheet(CString tempName);

BOOL GetTempSheet(CAdoInterface& myado, CString tempName, CStringArray& TempSheets);

void ExisttempSheet(int SheetChoose, BOOL Enable);

public:

// 数据库智能过滤重号数据

BOOL DBIFilterCheck;

afx\_msg void OnBnClickedButton13();

// 重复打标，不管重复性

BOOL RemarkCheck;

// 没有输入数据

BOOL EmptyDataCheck;

afx\_msg void OnBnClickedButton14();

//语音播报

HRESULT hr;

ISpVoice\* pVoice;

void Voice\_Ini();

void Voice\_Speak(CString Text);

public:

CString Hostname;

CString Ipaddress;

BOOL GetIp\_Name(CString& m\_hostname, CString& m\_ipaddress);

// 关联数据库获取数据

public:

CString IMEI\_MultNext[MULTDATAMAX][ARRAYMAXLENGTH];

public:

CString IMEI\_Mult[MULTDATAMAX];

int MultCount;

BOOL DataRelativeChoose;

public:

BOOL GetDataFromDB(CString FirstScan, int DataCount);

BOOL ConnecDB(CString m\_server, CString m\_db, CString m\_user, CString m\_pwd);

afx\_msg void OnBnClickedCheck9();

public:

CStringArray strdataArray; //数据字符串一维数组

// 还未使用的数据

CComboListCtrl Excell\_Unused;

afx\_msg void OnBnClickedButton18();

BOOL ListOutData();

void ListNoOutData();

CString RelativeField;

};

#pragma once

#include "AdoInterface.h"

//#include "md5.h"

#include "afxwin.h"

// CUserSetting 对话框

class CUserSetting : public CDialog

{

DECLARE\_DYNAMIC(CUserSetting)

public:

CUserSetting(CWnd\* pParent = NULL); // 标准构造函数

virtual ~CUserSetting();

// 对话框数据

enum { IDD = IDD\_USERSETTING };

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持

DECLARE\_MESSAGE\_MAP()

public:

BOOL OnInitDialog();

CString GetData(char\* Serial\_Order\_Return, CString Start, CString End, int Count = 1, int HandleNum = 0);//获取有效数据

afx\_msg void OnTimer(UINT nIDEvent);//定时器

public:

CAdoInterface myado;

CString LimitsSys;

public:

BOOL ConnecDB(CString m\_server, CString m\_db, CString m\_user, CString m\_pwd);

BOOL GetUserType(CAdoInterface& myado, CString Name, CString Password, CString \*Limits);

BOOL UpdateUserToDb(CString Name, CString Password, CString Limits,BOOL ChangePw=FALSE, CString NewPassword="");

BOOL DeleteUserFromDb(CString Name, CString Password,BOOL UsePW=TRUE);

public:

CStringArray UserArray;

public:

BOOL GetUsersFromDB();

void ChangeCombo();

BOOL CheckUser(CString Name, CString Password, CString\* Limits);

public:

afx\_msg void OnBnClickedButton1();

// 用户

CString UserName;

CString Password;

CString NewPassword;

CString Limits; //权限

afx\_msg void OnBnClickedButton7();

afx\_msg void OnBnClickedButton6();

afx\_msg void OnBnClickedButton10();

CComboBox LimitsControl;

afx\_msg void OnBnClickedButton11();

afx\_msg void OnCbnSelchangeCombo2();

CComboBox AllUsersBox;

};

// CLogin 对话框

class CLogin : public CDialog

{

DECLARE\_DYNAMIC(CLogin)

public:

CLogin(CWnd\* pParent = NULL); // 标准构造函数

virtual ~CLogin();

// 对话框数据

enum { IDD = IDD\_LOGIN };

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持

DECLARE\_MESSAGE\_MAP()

public:

// 登陆账号

CString m\_s\_name;

CString m\_s\_pwd;

afx\_msg void OnBnClickedOk();

void\* pdlgPointer;

};

#pragma once

#include "ComboListCtrl.h"

#include "afxwin.h"

#define DATA\_ITEM\_MAX 48

//#include "g:\pc备份\bartendertest\_plc\comboxlist\combolistctrl.h"

// CPrintBatch 对话框

#include "winspool.h"

#include "BartenderTestDlg.h"

class CPrintBatch : public CDialog

{

DECLARE\_DYNAMIC(CPrintBatch)

public:

CPrintBatch(CWnd\* pParent = NULL); // 标准构造函数

virtual ~CPrintBatch();

// 对话框数据

enum { IDD = IDD\_DIALOGCOLORBOX };

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持

virtual BOOL PreTranslateMessage(MSG\* pMsg);

void OnOK();//VK\_RETURN

virtual BOOL OnInitDialog();

afx\_msg void OnTimer(UINT nIDEvent);

DECLARE\_MESSAGE\_MAP()

public:

afx\_msg void OnEnChangeScandata();

CComboListCtrl m\_list2;

CComboBox PrinterName;

CString PrinterNameCS;

public:

CEdit PrintStateShow;

CString PrintTemplateCS;

public:

//打印机

CBtApplication btApp[2]; //DBtApplication与IBtApplication都有quit函数

CBtFormat btFormat[2];

CBtFormats\* btFormats[2];

BOOL PrinterIniDone[2];

int PrinterNum;

public:

CAdoInterface myado;

//BOOL UserLimit;

public:

void EnableWindowControl(BOOL Choose=TRUE);

void Delete\_Running\_Bartender();

void ListProcesses();//列举进程，然后结束Bartender.exe

BOOL GetSettingFromDB(CString m\_server,CString m\_db,CString m\_user,CString m\_pwd,CString ZhiDanNum);

public:

void LogShow\_exchange(CString Msg\_Log);

\_\_int64 ComputeCD(\_\_int64 nImei);

BOOL CheckIMEI(CString m\_IMEI);

public:

BOOL GetDataFromDB(CString m\_server,CString m\_db,CString m\_user,CString m\_pwd,CString FirstScan);

void PrintFunction();

public:

CString IMEI\_Mult[MULTDATAMAX];

int MultCount;

CComboBox MultScanControl;

BOOL CheckIMEIData;

CString DataPrintInput;

CEdit DataPrintInputControl;

afx\_msg void OnBnClickedButton6();

BOOL DataRelativeChoose;

};

---------------------------以上为代码前三十页---------------------------

---------------------------以下为代码后三十页---------------------------

// Rework.cpp : 实现文件

//

#include "stdafx.h"

#include "BartenderTest.h"

#include "Rework.h"

#include "BartenderTestDlg.h"

// CRework 对话框

IMPLEMENT\_DYNAMIC(CRework, CDialog)

CRework::CRework(CWnd\* pParent /\*=NULL\*/)

: CDialog(CRework::IDD, pParent)

, BoxNumber(\_T(""))

, ZhiDanValue(\_T(""))

{

}

CRework::~CRework()

{

}

void CRework::DoDataExchange(CDataExchange\* pDX)

{

CDialog::DoDataExchange(pDX);

DDX\_Text(pDX, IDC\_EDIT1, BoxNumber);

DDX\_Control(pDX, IDC\_EDIT1, BoxNumberControl);

DDX\_Control(pDX, IDC\_RESULT1, m\_Result1);

DDX\_Text(pDX, IDC\_EDIT2, ZhiDanValue);

}

BEGIN\_MESSAGE\_MAP(CRework, CDialog)

ON\_WM\_CLOSE()

ON\_EN\_CHANGE(IDC\_EDIT2, &CRework::OnEnChangeEdit2)

ON\_EN\_CHANGE(IDC\_EDIT1, &CRework::OnEnChangeEdit1)

END\_MESSAGE\_MAP()

// CRework 消息处理程序

void CRework::OnEnChangeEdit2()

{

UpdateData(TRUE);

}

void CRework::LogShow\_exchange(CEdit\* m\_Result,CString Msg\_Log,int HandleNum)

{

CString LOG\_Show\_Old;

m\_Result->GetWindowText(LOG\_Show\_Old);

CTime cTime;

cTime = CTime::GetCurrentTime(); //创建以当日日期的文件夹来保存数据

CString LogTime;

LogTime.Format(" (%04d-%02d-%02d--%02d:%02d:%02d)",cTime.GetYear(), cTime.GetMonth(), cTime.GetDay(),cTime.GetHour(), cTime.GetMinute(), cTime.GetSecond());

m\_Result->SetWindowText(Msg\_Log+LogTime+"\r\n"+LOG\_Show\_Old.Left(8192));

m\_Result->UpdateWindow();

}

BOOL CRework::Data\_DB\_CheckDelete(int SheetChoose)

{

CString Conn="";

CString DB\_FAIL="";

Conn.Format("driver={SQL Server};Server=%s;DATABASE=%s;UID=%s;PWD=%s", m\_server,m\_db,m\_user, m\_pwd);

if(myado[SheetChoose].m\_pCon==NULL)

{

CoInitialize(NULL);

DB\_FAIL=myado[SheetChoose].ConnecDataLibrary(Conn,"","",adModeUnknown);

}

if(myado[SheetChoose].m\_pCon->State==0) //1表示已经打开,0表示关闭，数据库意外断开，重连

{

LogShow\_exchange(&m\_Result1,"IMEI数据库意外断开，重连........",0);

DB\_FAIL=myado[SheetChoose].ConnecDataLibrary(Conn,"","",adModeUnknown);//数据库意外断开，重连

}

else

{

LogShow\_exchange(&m\_Result1,"数据库连接正常中........",0);

}

if(DB\_FAIL=="FAIL")

{

LogShow\_exchange(&m\_Result1,"连接数据库失败，无法上传数据，请检查网络.......",0);

return FALSE;

}

BOOL TestedSign=FALSE;

if((BoxNumber!="")&&(ZhiDanValue!=""))

{

BOOL UP\_Barcode=TRUE;

CString strSQL1;

strSQL1="DELETE from dbo.Gps\_CartonBoxTwenty\_Result WHERE BoxNo='"+BoxNumber+"' AND ZhiDan='"+ZhiDanValue+"'";

\_variant\_t var;

UP\_Barcode=myado[SheetChoose].Execute(strSQL1,&var);

if(UP\_Barcode==TRUE)

{

CString UpLoadNum;

UpLoadNum.Format("%d",var.intVal);

LogShow\_exchange(&m\_Result1,"成功删除记录数量: "+UpLoadNum,0);

}

else

{

LogShow\_exchange(&m\_Result1,"返工失败！！！！！！！！！！",0);

}

}

else

{

AfxMessageBox("制单号或者箱号不能为空！！");

return FALSE;

}

return TRUE;

}

void CRework::OnEnChangeEdit1()

{

UpdateData(TRUE);

if(((BoxNumber.Find("\r")!=-1)|| (BoxNumber.Find("\n")!=-1)) )//选择了IMEI1 &&IMEI\_length==15

{

BoxNumberControl.EnableWindow(FALSE);

//check whether digits

BoxNumber.Replace("\r","");

BoxNumber.Replace("\n","");

Data\_DB\_CheckDelete(1);

UpdateData(FALSE);//将输入的回车清掉

BoxNumberControl.EnableWindow(TRUE);

BoxNumberControl.SetFocus();

BoxNumberControl.SetSel(0,-1);

}

}

BOOL CRework::PreTranslateMessage(MSG\* pMsg)

{

// TODO: Add your specialized code here and/or call the base class

if(WM\_KEYDOWN == pMsg->message )

{

UINT nKey = (int) pMsg->wParam;

if( VK\_ESCAPE == nKey )

{

return TRUE;

}

}

return CDialog::PreTranslateMessage(pMsg);

}

void CRework::OnOK()

{

return;

CDialog::OnOK();

}

#pragma once

#include <atltime.h>

#include "ISocketStream.h"

#include <WS2tcpip.h>

#pragma comment(lib, "Ws2\_32.lib")

/////////////////////////////////////////////////////////////////////////////

// CActiveSock

class CActiveSock : public ISocketStream

{

public:

CActiveSock(HANDLE StopEvent);

virtual ~CActiveSock();

bool Connect(LPCTSTR HostName, USHORT PortNumber, CString IPAddr);

void SetRecvTimeoutSeconds(int NewTimeoutSeconds);

int GetRecvTimeoutSeconds();

void SetSendTimeoutSeconds(int NewTimeoutSeconds);

int GetSendTimeoutSeconds();

// Receives up to Len bytes of data and returns the amount received - or SOCKET\_ERROR if it times out

int RecvPartial(LPVOID lpBuf, const ULONG Len);

// Sends up to Len bytes of data and returns the amount sent - or SOCKET\_ERROR if it times out

int SendPartial (LPCVOID lpBuf, const ULONG Len);

// Receives exactly Len bytes of data and returns the amount sent - or SOCKET\_ERROR if it times out

int RecvMsg (LPVOID lpBuf, const ULONG Len);

// Sends exactly Len bytes of data and returns the amount sent - or SOCKET\_ERROR if it times out

int SendMsg(LPCVOID lpBuf, const ULONG Len);

BOOL ShutDown(int nHow = SD\_BOTH);

DWORD GetLastError();

bool Close(void); // Returns true if the close worked

private:

CTime RecvEndTime;

CTime SendEndTime;

static WSADATA WsaData;

WSAEVENT write\_event;

WSAEVENT read\_event;

WSAOVERLAPPED os;

bool RecvInitiated;

SOCKET ActualSocket;

DWORD LastError;

int SendTimeoutSeconds, RecvTimeoutSeconds;

HANDLE m\_hStopEvent;

};

// BartenderTestDlg.h : 头文件

//

#pragma once

#include "afxwin.h"

#define DATA\_NUM\_MAX 320 //一箱里终端数量

//#define LOCAL\_IMEI\_MAX 20 //箱数量//#define LOCAL\_IMEI\_MAX 2 //箱数量调试验证

#define SOFTVERSION\_LINE 5 //5

#define SETTING\_ITEM\_MAX 16

#define DATA\_ITEM\_MAX 48 //一个制单的配置项最多48

#define MULTDATAMAX 13 //一次最大输入数据量

#define PORTS\_NUM 8 //最大串口数量

#define THREAD\_NUM 4

#include "IniFile.H"

#define STRING\_PATH\_CONFIG "config\\Seting\_CONFIG.dll"

#define STRING\_PATH\_CONFIGLanguage "config\\Seting\_CONFIGLanguage.dll"

#define STRING\_SECTION\_CONFIGLanguage "SETTINGLanguage"

#define RESTARTSCAN 50

#include "AdoInterface.h"

//改变窗口大小

#include "ResizableDialog.h"

//列表

#include "combolistctrl.h"

#include ".\comboxlist\combolistctrl.h"

//获取打印机

#include "winspool.h"

#include "CBtApplication.h"

#include "CBtFormat.h"

#include "CBtFormats.h"

//继电器控制

#include "Modbus.h"

#include "MbCommon.h"

//语音播报

#include <sapi.h>

//获取系统运行的进程

#include <tlhelp32.h>

#include "WorkQueue.h"

#define MAXPAGE 8

typedef struct

{

LPVOID WinHandle;

int Count;

CBtFormat btFormatCurrent;

}

PARA\_Handle;

//多线程获取端口

static CRITICAL\_SECTION GETPORTCR;

// SpecificWorkItem 类

class SpecificWorkItem : public WorkItemBase //新建一个基于队列的基类的子类，该子类的私有函数能被CWorkQueue调用

{

public:

SpecificWorkItem(char \*defaultKeyFile, char \*defaultDbFile, char \*outFileName, int nItem, LPVOID QueueHandle)

{

defaultKeyFile\_M = defaultKeyFile; defaultDbFile\_M = defaultDbFile; outFileName\_M = outFileName; m\_nItem = nItem;

QueueHandle\_M = QueueHandle;

}

private:

void DoWork(int strCommand\_Head\_Label);

void Abort();

int m\_nItem;

char \*defaultKeyFile\_M;

char \*defaultDbFile\_M;

char \*outFileName\_M;

LPVOID QueueHandle\_M;

};

// CBartenderTestDlg 对话框

class CBartenderTestDlg : public CResizableDialog

{

// 构造

friend CWorkQueue; //对外部友元类的声明，可以访问类CBartenderTestDlg 的 私有成员

public:

CBartenderTestDlg(CWnd\* pParent = NULL); // 标准构造函数

CWorkQueue m\_WorkQueue; //重载队列类

public:

static unsigned long \_\_stdcall ThreadFuncDestroy(void\* pParam);//销毁线程

UINT pRunComandThread(int pParam, CString FirstInput, CString ReplaceValue);//命令解析的线程函数

void StopWorkItem();//停止运行

public:

// 对话框数据

enum { IDD = IDD\_BARTENDERTEST\_DIALOG };

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持

afx\_msg LRESULT PopulateComboList2(WPARAM wParam, LPARAM lParam);//在list控件上增加combo box控件

afx\_msg LRESULT OnEndLabelEditVariableCriteria(WPARAM wParam, LPARAM lParam);

public:

void CloseFunc();

// 实现

protected:

HICON m\_hIcon;

// 生成的消息映射函数

virtual BOOL OnInitDialog();

afx\_msg HBRUSH OnCtlColor(CDC\* pDC,CWnd\* pWnd,UINT nCtlColor);

afx\_msg void OnTimer(UINT nIDEvent);

virtual BOOL PreTranslateMessage(MSG\* pMsg);

virtual void OnOK();

afx\_msg void OnPaint();

afx\_msg HCURSOR OnQueryDragIcon();

DECLARE\_MESSAGE\_MAP()

public:

void LogShow\_exchange(CEdit\* m\_Result,CEdit\* Final\_Result\_Control,int State,CString Msg\_Log,int HandleNum);

\_\_int64 ComputeCD(\_\_int64 nImei);

BOOL CheckIMEI(CString m\_IMEI);

BOOL CheckData(int IMEI\_Count);

public:

COLORREF gColor;

CString LocalCurrentIMEI;

CString LocalCurrentSN;

int LOCAL\_IMEI\_MAX;

CString\* LocalIMEI;

CString\* LocalSN;

//CString LocalIMEI[LOCAL\_IMEI\_MAX];

//CString LocalSN[LOCAL\_IMEI\_MAX];

int Box\_Count;

BOOL RestartAll;

BOOL AskOnce[5];

BOOL CloseApp;

BOOL DeleteTempSheet;

CString Async\_IMEI\_Input[DATA\_NUM\_MAX];

CString Async\_Other\_Input[DATA\_NUM\_MAX];

int Async\_IMEI\_Count;

CString Async\_IMEI\_Count\_CS;

CString IMEI\_Mult[MULTDATAMAX];

CString IMEI\_Input[DATA\_NUM\_MAX];

CString SN\_Get[DATA\_NUM\_MAX];

CString Other\_Input[DATA\_NUM\_MAX];

int IMEI\_Count;

CString IMEI\_Count\_CS;

BOOL DataprintResult;

int DataprintResultCount;//关联输入、异步检查的时候，确保所有输入判断完成后，再运行扫描下一个条码

public:

BOOL GetDataFromDB(int SheetChoose,CString SN,CString IMEI);

public:

afx\_msg void OnBnClickedButton1();

afx\_msg void OnEnChangeDataprint();

CString DataPrintInput;

CEdit DataPrintInputControl;

CEdit Final\_Result\_Control1;

CEdit m\_Result1;

afx\_msg void OnBnClickedButton36();

// IMEIs显示

CEdit m\_Result2;

// Tac信息

CString IMEITac;

CEdit IMEITacControl;

CString IMEIEnd;

CEdit IMEIEndControl;

public:

void LoadConfigLanguage();

void SaveConfigLanguage();

void GetSetting();//读取设置项目

void LoadConfig();

void SaveConfig();

BOOL Data\_Update(int SheetChoose);

BOOL Data\_DB\_CheckIni(int SheetChoose,BOOL Enable=FALSE,BOOL DeleteTemp=FALSE);

BOOL Data\_DB\_Check(int SheetChoose, CString IMEIData, CString OtherData = "");

BOOL Data\_DB\_CheckBoxNo(int SheetChoose);

CString GetSN(int SheetChoose,CString IMEIData);

BOOL CheckDataDB(int SheetChoose, BOOL Restart, CString IMEIData, CString OtherData="");

void Async\_CheckFunction(int Async\_IMEI\_Count);

void ClearLocalIMEICache();

BOOL LocalIMEICache();

int getCountStr(CString strLongstr, CString strFindstr);

BOOL CheckDataCache(CString IMEIData,BOOL IMEIorSN=TRUE,BOOL AddData=TRUE);

//打印处理

void GetPrinters();

public:

void EnableWindowControl(BOOL Choose=TRUE);

void LockComPortControl(BOOL Choose=TRUE);

void LessWindowControl(BOOL Choose=TRUE);

public:

CWinThread\* Test\_Handle;

CWinThread\* GetComData\_Handle;

BOOL GetComDataEnable;

BOOL GetComDataEnable1;

BOOL GetComDataEnable2;

int RestartScanCount;

public:

CAdoInterface myado[5]; //myado[0]--->测试表;myado[1]--->装箱表;myado[2]--->关联数据表;myado[3]--->获取配置表;myado[4]--->上传错误信息

CString WetherUpdateData\_M; //数据是否上传数据库

CString WeightConfirmTime\_M;

CString WeightDiff\_M;

CString WeightCenter\_M;

CString WeightLimit\_M;

CString PneumaticDuration\_M;

CString RelayAddr\_M;

CString RelayEquipType\_M;

CString m\_server\_M;

CString m\_db\_M ;

CString m\_user\_M;

CString m\_pwd\_M ;

CString STRING\_SECTION\_CONFIG;

CString ProductLanguage;

CString ProductCount;

CString BoxCountRefresh;//N箱后刷新一次数据

CString MachineType;

CString ProductColor;

CString ProductWeight;

CString ProductDate;

CString ProductNum;

CString Software\_Version;

CString BoxNum;

CString ZhiDan;

CString Remark;

CString SNNumber; //SN号码

/////////////////////////////////////////////////

CString QRCodeValueWithoutR; //二维码变量

CString QRCodeValueWithoutR\_T[MAXPAGE]; //二维码分页变量

CString QRCodeValueDB;

CString QRCodeValue; //TXT变量

CString ZhiDanValue; //制单

CString MachineTypeValue; //变量值

CString ProductColorValue;

CString ProductWeightValue;

CString ProductDateValue;

CString ProductNumValue;

CString Software\_VersionValue;

CString BoxNumValue;

CString RemarkValue;

CString ProductCountValue;

CString ProductCountValueDBUpdate;

CString TextMachineType;//文本

CString TextProductColor;

CString TextProductWeight;

CString TextProductDate;

CString TextProductNum;//产品号

CString TextSoftware\_Version;

CString TextBoxNumOnly;

CString TextBoxNum;

CString TextZhiDan;

CString TextRemark;

CString TextQRCode;

CString TextProductCount;//产品数量

CString All\_Setting[SETTING\_ITEM\_MAX];

//上一次变量

CString IMEIStartLastOne;

CString IMEIEndLastOne;

CString Software\_VersionLastOne;

CString Software\_VersionNew;

BOOL ParaDownLoad\_CheckLastOne;

BOOL Function\_CheckLastOne;

BOOL Couple\_CheckLastOne;

BOOL WriteData\_CheckLastOne;

BOOL CartonSta\_CheckLastOne;

BOOL LeaveOutSta\_CheckLastOne;

public:

CString IMEIStart;

CEdit IMEIStartControl;

afx\_msg void OnBnClickedButton2();

// 异步检查

BOOL Async\_Check;

afx\_msg void OnBnClickedCheck1();

// 数据库异常IMEI记录

CEdit m\_ResultError;

// 一箱数量

CComboBox NumberSelect;

CComboListCtrl m\_list2;

public:

BOOL LockState;

public:

afx\_msg void OnBnClickedButton4();

afx\_msg void OnBnClickedButton3();

// 语言选择

CComboBox LanguageSelect;

afx\_msg void OnCbnSelchangeCombo2();

afx\_msg void OnBnClickedButton5();

//参数下载工位检查

BOOL ParaDownLoad\_Check;

CButton ParaDownLoad\_CheckControl;

afx\_msg void OnBnClickedButton6();

CComboBox PrinterName;

CComboBox PrinterName2;

CString PrinterNameCS;

CString PrinterNameCS2;

afx\_msg void OnBnClickedButton7();

// 打印模板1

CString PrintTemplateCS;

CString PrintTemplateCS2;

afx\_msg void OnBnClickedButton10();

BOOL PrinterChooseCS;

BOOL PrinterChooseCS2;

BOOL LeaveOutSta\_Check;

BOOL CartonSta\_Check;

public:

//打印机

CBtApplication btApp[3]; //DBtApplication与IBtApplication都有quit函数

CBtFormat btFormat[3];

CBtFormats\* btFormats[3];

BOOL PrinterIniDone[3];

int PrinterNum;

int FirstScanResult;//第一次扫描结果

public:

BOOL PrintFunction1();

BOOL PrintFunction2();

BOOL OtherDataInputFunction();

void SaveConfigChange();

public:

CFile cReport[2];

public:

void WriteLogIni(int LogNum);//写LOG信息

void WriteLog(CString Text,int LogNum);

void WriteLogEnd(int LogNum);

public:

afx\_msg void OnBnClickedButton8();

afx\_msg void OnBnClickedButton9();

afx\_msg void OnBnClickedButton11();

afx\_msg void OnBnClickedButton12();

public:

int CurreScanTime;

// 其它参数(SN/手机号等)

CString OtherParaCS;

CString OtherParaValueCS;

CString OtherParaLengthCS;

CString OtherParaPrefixCS;

afx\_msg void OnBnClickedButton13();

CEdit OtherParaValueControl;

afx\_msg void OnBnClickedButton14();

// 检查IMEI重复性的选择变更

CButton CartonSta\_CheckControl;

CButton LeaveOutSta\_CheckControl;

public:

//CClientComm RelyComm;//继电器控制通信

BOOL COM\_IniDone;

CString COM\_Number[PORTS\_NUM];

HANDLE hPort[THREAD\_NUM];

// 继电器控制端口

CComboBox CtrComport;

CComboBox CtrBaud;

public:

BOOL COM\_Opened;

public:

u8 equipaddr; //设备地址

public:

char HexChar(char c);

int Str2Hex(CString str, char\* data);

s16 OpertorDO(s16 ioindex,bool onoff); //处理单个继电器

s16 OpertorStep(s16 ioindex,bool onoff);//控制步进电机

public:

BOOL StringToHexGroup(unsigned char \*OutHexBuffer, char \*InStrBuffer, unsigned int strLength);//string转HEX "12AB"--->0x12 0xab

char HexToASCII(unsigned char data\_hex);

void HexGroupToString(char \*OutStrBuffer, unsigned char \*InHexBuffer, unsigned int HexLength);//用于显示串口返回值

unsigned int SeperateToHex(const char \*src, unsigned char \*des); //"12 AB"--->0x120xAB--->兼容空格

char\* AscToHexString(char\* pInData, char\* pOutData); //"ab01234" -> "61 62 30 31 32 33 34 "

public:

afx\_msg void OnBnClickedButton15();

void InitCOM(CComboBox\* m\_Port,CComboBox\* m\_Baud,int num);

BOOL OnGetport(int HandleNum);

BOOL OPen\_Serial\_Port(CComboBox\* m\_Port,CComboBox\* m\_Baud,int HandleNum);

char\* Send\_Serial\_Order(CString\* Vaule\_Return,CString strCommand\_Vaule,u8\* HEXData,int HandleNum,char\* EndSign);

bool IMEI\_Function\_Judge(int i,CString IMEI\_FT\_Item,char\* Serial\_Order\_Return,int HandleNum);

BOOL WriteAT\_Thread(CComboBox\* m\_Port,CComboBox\* m\_Baud,int HandleNum,CString OrderData,u8\* HEXData,char\* EndSign="NULL");

afx\_msg void OnBnClickedButton16();

afx\_msg void OnBnClickedButton17();

//语音播报

HRESULT hr;

ISpVoice\* pVoice;

void Voice\_Ini();

void Voice\_Speak(CString Text);

public:

// 串口扫描枪

CComboBox ScanGunPort1;

CComboBox ScanGunBaud1;

HANDLE hScanGun[THREAD\_NUM];

CString Vaule\_ScanGun[THREAD\_NUM]; //扫描枪数据

BOOL Thread\_State[THREAD\_NUM];

BOOL Thread\_StateScanClear;

CString Port\_TempScan[THREAD\_NUM];//串口读取数据 的缓存

public:

BOOL WeightResult;

BOOL ScanStart;

public:

void Delay\_Clock(UINT mSecond);// 延时

void ComFuntion(int HandleNum);

void ComFuntion1(int HandleNum);

void ComFuntion2(int HandleNum);

BOOL OPen\_ScanGun\_Port(CComboBox\* m\_Port,CComboBox\* m\_Baud,int HandleNum); //打开扫描枪端口

BOOL Get\_ScanGun\_Data(int HandleNum,char\* EndSign); //获取扫描枪端口数据

BOOL Check\_ScanGun\_Data(int HandleNum,BOOL ImeiChoose,int LengthLimit);

BOOL Check\_Weight\_Data(int HandleNum,float Weight);

public:

void ErrorFunction();//继电器2开关

public:

int UnstandardWeight;

public:

afx\_msg void OnBnClickedButton18();

afx\_msg void OnBnClickedButton19();

// 设置路径

CString SettingPath;

afx\_msg void OnBnClickedButton20();

// 压力传感器端口

CComboBox PressComport;

CComboBox PressBaud;

afx\_msg void OnBnClickedButton21();

afx\_msg void OnBnClickedButton22();

// 校验IMEI

BOOL CheckIMEIData;

// 串口扫描枪2

CComboBox ScanGunPort2;

CComboBox ScanGunBaud2;

afx\_msg void OnBnClickedButton23();

afx\_msg void OnBnClickedButton24();

afx\_msg void OnBnClickedButton25();

afx\_msg void OnBnClickedButton26();

afx\_msg void OnBnClickedButton27();

afx\_msg void OnBnClickedButton28();

afx\_msg void OnBnClickedButton29();

afx\_msg void OnBnClickedButton30();

afx\_msg void OnBnClickedButton31();

afx\_msg void OnBnClickedButton32();

afx\_msg void OnBnClickedButton33();

// 分几页打印

CString PageNum;

CComboBox PageNumControl;

// 一台机IMEI号的数量

CString MultIMEI;

CComboBox MultIMEIControl;

// 数据关联获取

BOOL DataRelative;

void EnableFRDC(BOOL UseQueue = FALSE,BOOL ENABLE=TRUE);

void EnableFRDCQueue(BOOL UseQueue = FALSE, BOOL ENABLE = TRUE);

void QueueFunc(CString FirstInput, CString ReplaceValue);

afx\_msg void OnEnChangeEdit15();

// 选择数据关联时，第一个输入数据

CString FirstRelativeData;

CEdit FirstRelativeDataControl;

public:

void ListProcesses();//列举进程，然后结束Bartender.exe

BOOL GetSettingFromDB(CString m\_server,CString m\_db,CString m\_user,CString m\_pwd,CString ZhiDanNum);

afx\_msg void OnEnChangeEdit16();

CEdit InputZhidan;

CString InputZhidanCS;

// 数据库漏测高级选择

BOOL Function\_Check;

CButton Function\_CheckControl;

BOOL Couple\_Check;

CButton Couple\_CheckControl;

BOOL WriteData\_Check;

CButton WriteData\_CheckControl;

// 功能漏测数据查询进行时间过滤

CComboBox LeaveOutSta\_TimeFilter;

CComboBox StationsChoose;

CString StationsChooseCS;

//清理临时表

BOOL GetTempSheet(CAdoInterface& myado, CString tempName, CStringArray& TempSheets);

BOOL DeletetempSheet(int SheetChoose, CString tempName);

void ExisttempSheet(int SheetChoose, BOOL Enable);

afx\_msg void OnBnClickedButton35();

// 返工编码

CComboBox ReturnCodeControl;

CString ReturnCodeCS;

// 使用队列

BOOL RelativeQueue;

public:

CString Hostname;

CString Ipaddress;

BOOL GetIp\_Name(CString& m\_hostname, CString& m\_ipaddress);

afx\_msg void OnEnChangeEdit18();

// 根据箱号获取数据库数据

CString BoxNumDBData;

CEdit BoxNumDBDataControl;

BOOL GetBoxDataFromDB(int SheetChoose, CString BoxNumCS);

BOOL PrintFunction2DB();

BOOL PrintFunction3DB();

BOOL PrintFunction3();

// 是否检查箱号

BOOL CartonBox\_Check;

public:

BOOL Data\_UpdateError(CAdoInterface& myado, int DataUpNum, CEdit\* m\_Result, CEdit\* Final\_Result\_Control, CString Message, CString Category, CString ProductID);

BOOL GetExistError(CAdoInterface& myado, int DataUpNum, CEdit\* m\_Result, CEdit\* Final\_Result\_Control, CString \*ExistMessage, CString ProductID);

// 打印机3

CComboBox PrinterName3;

CString PrinterNameCS3;

BOOL PrinterChooseCS3;

CString PrintTemplateCS3;

afx\_msg void OnBnClickedButton38();

afx\_msg void OnBnClickedButton39();

afx\_msg void OnBnClickedButton37();

// 卡通箱贴纸包含软件版本

BOOL IncludeSWChooseCS;

BOOL CheckZhiDanChooseCS;

afx\_msg void OnCbnSelchangeCombo7();

// 其他输入的替换值

BOOL OtherReplace;//控件变量

BOOL OtherStart;

CString OtherReplaceValue;

// 关联字段

CString RelativeField;

// 输入替代的截取长度

CString ReplaceLength;

// 输入递增选择

BOOL InputIncrease;

// 检查关联表绑定状态

BOOL CheckBindState;

CString ReplaceCheckData;

CString CutDataFormat;

};

// 从类型库向导中用“添加类”创建的计算机生成的 IDispatch 包装类

#import "C:\\Program Files\\Seagull\\BarTender Suite\\bartend.exe" no\_namespace

// CBtApplication 包装类

class CBtApplication : public COleDispatchDriver

{

public:

CBtApplication(){} // 调用 COleDispatchDriver 默认构造函数

CBtApplication(LPDISPATCH pDispatch) : COleDispatchDriver(pDispatch) {}

CBtApplication(const CBtApplication& dispatchSrc) : COleDispatchDriver(dispatchSrc) {}

// 属性

public:

// 操作

public:

// IBtApplication 方法

public:

LPDISPATCH get\_ActiveFormat()

{

LPDISPATCH result;

InvokeHelper(0x1, DISPATCH\_PROPERTYGET, VT\_DISPATCH, (void\*)&result, NULL);

return result;

}

long get\_BuildNumber()

{

long result;

InvokeHelper(0x2, DISPATCH\_PROPERTYGET, VT\_I4, (void\*)&result, NULL);

return result;

}

LPDISPATCH get\_Formats()

{

LPDISPATCH result;

InvokeHelper(0x3, DISPATCH\_PROPERTYGET, VT\_DISPATCH, (void\*)&result, NULL);

return result;

}

BOOL get\_IsPrinting()

{

BOOL result;

InvokeHelper(0x4, DISPATCH\_PROPERTYGET, VT\_BOOL, (void\*)&result, NULL);

return result;

}

BOOL get\_IsProcessingCommandLines()

{

BOOL result;

InvokeHelper(0x5, DISPATCH\_PROPERTYGET, VT\_BOOL, (void\*)&result, NULL);

return result;

}

CString get\_Version()

{

CString result;

InvokeHelper(0x6, DISPATCH\_PROPERTYGET, VT\_BSTR, (void\*)&result, NULL);

return result;

}

void put\_Visible(BOOL newValue)

{

static BYTE parms[] = VTS\_BOOL ;

InvokeHelper(0x7, DISPATCH\_PROPERTYPUT, VT\_EMPTY, NULL, parms, newValue);

}

BOOL get\_Visible()

{

BOOL result;

InvokeHelper(0x7, DISPATCH\_PROPERTYGET, VT\_BOOL, (void\*)&result, NULL);

return result;

}

LPDISPATCH get\_Window()

{

LPDISPATCH result;

InvokeHelper(0x8, DISPATCH\_PROPERTYGET, VT\_DISPATCH, (void\*)&result, NULL);

return result;

}

void Quit(long SaveChanges)

{

static BYTE parms[] = VTS\_I4 ;

InvokeHelper(0x9, DISPATCH\_METHOD, VT\_EMPTY, NULL, parms, SaveChanges);

}

void Save(BOOL DoPrompt)

{

static BYTE parms[] = VTS\_BOOL ;

InvokeHelper(0xa, DISPATCH\_METHOD, VT\_EMPTY, NULL, parms, DoPrompt);

}

void put\_SAPIDocDefinitionFile(LPCTSTR newValue)

{

static BYTE parms[] = VTS\_BSTR ;

InvokeHelper(0xc, DISPATCH\_PROPERTYPUT, VT\_EMPTY, NULL, parms, newValue);

}

CString get\_SAPIDocDefinitionFile()

{

CString result;

InvokeHelper(0xc, DISPATCH\_PROPERTYGET, VT\_BSTR, (void\*)&result, NULL);

return result;

}

void CommandLine(LPCTSTR CommandLine)

{

static BYTE parms[] = VTS\_BSTR ;

InvokeHelper(0xb, DISPATCH\_METHOD, VT\_EMPTY, NULL, parms, CommandLine);

}

LPDISPATCH get\_LicenseServer()

{

LPDISPATCH result;

InvokeHelper(0xd, DISPATCH\_PROPERTYGET, VT\_DISPATCH, (void\*)&result, NULL);

return result;

}

CString XMLScript(LPCTSTR XMLScript, long SourceType, LPDISPATCH \* Messages)

{

CString result;

static BYTE parms[] = VTS\_BSTR VTS\_I4 VTS\_PDISPATCH ;

InvokeHelper(0x11, DISPATCH\_METHOD, VT\_BSTR, (void\*)&result, parms, XMLScript, SourceType, Messages);

return result;

}

CString get\_Edition()

{

CString result;

InvokeHelper(0x14, DISPATCH\_PROPERTYGET, VT\_BSTR, (void\*)&result, NULL);

return result;

}

void put\_VisibleWindows(long newValue)

{

static BYTE parms[] = VTS\_I4 ;

InvokeHelper(0x16, DISPATCH\_PROPERTYPUT, VT\_EMPTY, NULL, parms, newValue);

}

long get\_VisibleWindows()

{

long result;

InvokeHelper(0x16, DISPATCH\_PROPERTYGET, VT\_I4, (void\*)&result, NULL);

return result;

}

void put\_TopMost(BOOL newValue)

{

static BYTE parms[] = VTS\_BOOL ;

InvokeHelper(0x17, DISPATCH\_PROPERTYPUT, VT\_EMPTY, NULL, parms, newValue);

}

BOOL get\_TopMost()

{

BOOL result;

InvokeHelper(0x17, DISPATCH\_PROPERTYGET, VT\_BOOL, (void\*)&result, NULL);

return result;

}

void put\_ParentWindow(unsigned long newValue)

{

static BYTE parms[] = VTS\_UI4 ;

InvokeHelper(0x18, DISPATCH\_PROPERTYPUT, VT\_EMPTY, NULL, parms, newValue);

}

unsigned long get\_ParentWindow()

{

unsigned long result;

InvokeHelper(0x18, DISPATCH\_PROPERTYGET, VT\_UI4, (void\*)&result, NULL);

return result;

}

long get\_ProcessId()

{

long result;

InvokeHelper(0x19, DISPATCH\_PROPERTYGET, VT\_I4, (void\*)&result, NULL);

return result;

}

CString get\_FullVersion()

{

CString result;

InvokeHelper(0x1a, DISPATCH\_PROPERTYGET, VT\_BSTR, (void\*)&result, NULL);

return result;

}

BOOL SpecifyPrintOnlyPassword(LPCTSTR Password)

{

BOOL result;

static BYTE parms[] = VTS\_BSTR ;

InvokeHelper(0x1b, DISPATCH\_METHOD, VT\_BOOL, (void\*)&result, parms, Password);

return result;

}

void AdministerMessageSetup()

{

InvokeHelper(0x1c, DISPATCH\_METHOD, VT\_EMPTY, NULL, NULL);

}

void AdministerAlertSetup()

{

InvokeHelper(0x1d, DISPATCH\_METHOD, VT\_EMPTY, NULL, NULL);

}

void AdministerLogSetup()

{

InvokeHelper(0x1e, DISPATCH\_METHOD, VT\_EMPTY, NULL, NULL);

}

unsigned long Private\_SetSharedMemory(LPCTSTR sharedMemoryName)

{

unsigned long result;

static BYTE parms[] = VTS\_BSTR ;

InvokeHelper(0x1f, DISPATCH\_METHOD, VT\_UI4, (void\*)&result, parms, sharedMemoryName);

return result;

}

void Private\_QueueCommandLine(LPCTSTR CommandLine, unsigned \_\_int64 cmdId)

{

static BYTE parms[] = VTS\_BSTR VTS\_UI8 ;

InvokeHelper(0x20, DISPATCH\_METHOD, VT\_EMPTY, NULL, parms, CommandLine, cmdId);

}

void SetGlobalDataFieldValue(LPCTSTR GlobalDataField, LPCTSTR Value)

{

static BYTE parms[] = VTS\_BSTR VTS\_BSTR ;

InvokeHelper(0x21, DISPATCH\_METHOD, VT\_EMPTY, NULL, parms, GlobalDataField, Value);

}

CString GetGlobalDataFieldValue(LPCTSTR GlobalDataField)

{

CString result;

static BYTE parms[] = VTS\_BSTR ;

InvokeHelper(0x22, DISPATCH\_METHOD, VT\_BSTR, (void\*)&result, parms, GlobalDataField);

return result;

}

// IBtApplication 属性

public:

};

// 从类型库向导中用“添加类”创建的计算机生成的 IDispatch 包装类

#import "C:\\Program Files\\Seagull\\BarTender Suite\\bartend.exe" no\_namespace

// CBtCache 包装类

class CBtCache : public COleDispatchDriver

{

public:

CBtCache(){} // 调用 COleDispatchDriver 默认构造函数

CBtCache(LPDISPATCH pDispatch) : COleDispatchDriver(pDispatch) {}

CBtCache(const CBtCache& dispatchSrc) : COleDispatchDriver(dispatchSrc) {}

// 属性

public:

// 操作

public:

// IBtCache 方法

public:

void put\_FlushInterval(long newValue)

{

static BYTE parms[] = VTS\_I4 ;

InvokeHelper(0x1, DISPATCH\_PROPERTYPUT, VT\_EMPTY, NULL, parms, newValue);

}

long get\_FlushInterval()

{

long result;

InvokeHelper(0x1, DISPATCH\_PROPERTYGET, VT\_I4, (void\*)&result, NULL);

return result;

}

void Flush()

{

InvokeHelper(0x2, DISPATCH\_METHOD, VT\_EMPTY, NULL, NULL);

}

// IBtCache 属性

public:

};

#pragma once

#include "afxwin.h"

// CIMEIPrint\_Setting 对话框

class CIMEIPrint\_Setting : public CDialog

{

DECLARE\_DYNAMIC(CIMEIPrint\_Setting)

public:

CIMEIPrint\_Setting(CWnd\* pParent = NULL); // 标准构造函数

virtual ~CIMEIPrint\_Setting();

// 对话框数据

enum { IDD = IDD\_SETTING };

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持

virtual BOOL OnInitDialog();//函数

afx\_msg void OnClose();

DECLARE\_MESSAGE\_MAP()

public:

void InitCOM(CComboBox\* m\_Port,CComboBox\* m\_Baud,int num);//初始化串口

BOOL OPen\_ScanGun\_Port(CComboBox\* m\_Port,CComboBox\* m\_Baud,int HandleNum);

void ComFuntion2(int HandleNum);

BOOL Get\_ScanGun\_Data(int HandleNum,char\* EndSign);

public:

BOOL COM\_IniDone;

CString COM\_Number[8];

HANDLE hScanGun[4];

CString Vaule\_ScanGun[4];

CString Port\_TempScan[4];//串口读取数据 的缓存

BOOL GetComDataEnable2;

public:

// 数据库服务器设置

CString m\_server;

CString m\_db;

CString m\_user;

CString m\_pwd;

// 继电器模块地址

CString RelayAddr;

// 设备型号

CString RelayEquipType;

// 称重合格范围

CString WeightLimit;

// 重量设置端口

CComboBox PressComportSet;

CComboBox PressBaudSet;

afx\_msg void OnBnClickedButton1();

afx\_msg void OnBnClickedButton2();

afx\_msg void OnBnClickedOk();

// 气动执行时长

CString PneumaticDuration;

// 称重中间值

CString WeightCenter;

CString WeightDiff;

afx\_msg void OnEnChangeEdit2();

afx\_msg void OnEnChangeEdit13();

// 称重确认次数

CString WeightConfirmTime;

CString WetherUpdateData;

};

#pragma once

class CEventWrapper

{

public:

CEventWrapper(LPSECURITY\_ATTRIBUTES lpEventAttributes = NULL,

BOOL bManualReset = TRUE,

BOOL bInitialState = FALSE,

LPCTSTR lpName = NULL)

: m\_Event(NULL)

{

m\_Event = CreateEvent(lpEventAttributes,bManualReset,bInitialState,lpName);

if (!m\_Event)

throw "no event";

}

HANDLE Event() const

{

return m\_Event;

}

operator const HANDLE()

{

return m\_Event;

}

~CEventWrapper()

{

if (m\_Event)

{

CloseHandle(m\_Event);

m\_Event = NULL;

}

}

private:

HANDLE

m\_Event;

};

// 从类型库向导中用“添加类”创建的计算机生成的 IDispatch 包装类

#import "C:\\Program Files\\Seagull\\BarTender Suite\\bartend.exe" no\_namespace

// CTemplateFieldDs 包装类

class CTemplateFieldDs : public COleDispatchDriver

{

public:

CTemplateFieldDs(){} // 调用 COleDispatchDriver 默认构造函数

CTemplateFieldDs(LPDISPATCH pDispatch) : COleDispatchDriver(pDispatch) {}

CTemplateFieldDs(const CTemplateFieldDs& dispatchSrc) : COleDispatchDriver(dispatchSrc) {}

// 属性

public:

// 操作

public:

// ITemplateFieldDs 方法

public:

// ITemplateFieldDs 属性

public:

CString GetName()

{

CString result;

GetProperty(0x1, VT\_BSTR, (void\*)&result);

return result;

}

void SetName(CString propVal)

{

SetProperty(0x1, VT\_BSTR, propVal);

}

CString GetValue()

{

CString result;

GetProperty(0x2, VT\_BSTR, (void\*)&result);

return result;

}

void SetValue(CString propVal)

{

SetProperty(0x2, VT\_BSTR, propVal);

}

CString GetFieldName()

{

CString result;

GetProperty(0x3, VT\_BSTR, (void\*)&result);

return result;

}

void SetFieldName(CString propVal)

{

SetProperty(0x3, VT\_BSTR, propVal);

}

};

#ifndef CSPREADSHEET\_H

#define CSPREADSHEET\_H

#include <odbcinst.h>

#include <afxdb.h>

class CSpreadSheet

{

public:

CSpreadSheet(CString File, CString SheetOrSeparator, bool Backup = true); // Open spreadsheet for reading and writing

~CSpreadSheet(); // Perform some cleanup functions

bool AddHeaders(CStringArray &FieldNames, bool replace = false); // Add header row to spreadsheet

bool DeleteSheet(); // Clear text delimited file content

bool DeleteSheet(CString SheetName); // Clear entire Excel spreadsheet content. The sheet itself is not deleted

bool AddRow(CStringArray &RowValues, long row = 0, bool replace = false); // Insert or replace a row into spreadsheet. Default is add new row.

bool AddCell(CString CellValue, CString column, long row = 0, bool Auto = true);

bool AddCell(CString CellValue, short column, long row = 0); // Replace or add a cell into spreadsheet using column number. Default is add cell into new row.

bool ReplaceRows(CStringArray &NewRowValues, CStringArray &OldRowValues); // Search and replace rows in Excel spreadsheet

bool ReadRow(CStringArray &RowValues, long row = 0); // Read a row from spreadsheet. Default is read the next row

bool ReadColumn(CStringArray &ColumnValues, CString column, bool Auto = true); // Read a column from Excel spreadsheet using header row or column alphabet. Set Auto to false if want to force column to be used as header name

bool ReadColumn(CStringArray &ColumnValues, short column); // Read a column from spreadsheet using column number

bool ReadCell (CString &CellValue, CString column, long row = 0, bool Auto = true);

bool ReadCell (CString &CellValue, short column, long row = 0); // Read a cell from spreadsheet using column number. Default is read the next cell in next row.

void BeginTransaction(); // Begin transaction

bool Commit(); // Save changes to spreadsheet

bool RollBack(); // Undo changes to spreadsheet

bool Convert(CString SheetOrSeparator);

inline void GetFieldNames (CStringArray &FieldNames) {FieldNames.RemoveAll(); FieldNames.Copy(m\_aFieldNames);} // Get the header row from spreadsheet

inline long GetTotalRows() {return m\_dTotalRows;} // Get total number of rows in spreadsheet

inline short GetTotalColumns() {return m\_dTotalColumns;} // Get total number of columns in spreadsheet

inline long GetCurrentRow() {return m\_dCurrentRow;} // Get the currently selected row in spreadsheet

inline bool GetBackupStatus() {return m\_bBackup;} // Get status of backup. True if backup is successful, False if spreadsheet is not backup

inline bool GetTransactionStatus() {return m\_bTransaction;} // Get status of Transaction. True if Transaction is started, False if Transaction is not started or has error in starting

inline CString GetLastError() {return m\_sLastError;} // Get last error message

CString m\_sSql; // SQL statement to open Excel spreadsheet for reading

CString m\_sDsn; // DSN string to open Excel spreadsheet for reading and writing

CString m\_stempSql; // Temporary string for SQL statements or for use by functions

CString m\_stempString; // Temporary string for use by functions

CString m\_sSheetName; // Sheet name of Excel spreadsheet

CString m\_sExcelDriver; // Name of Excel Driver

CString m\_sFile; // Spreadsheet file name

CString m\_sSeparator; // Separator in text delimited spreadsheet

CString m\_sLastError; // Last error message

CStringArray m\_atempArray; // Temporary array for use by functions

CStringArray m\_aFieldNames; // Header row in spreadsheet

CStringArray m\_aRows; // Content of all the rows in spreadsheet

CDatabase \*m\_Database; // Database variable for Excel spreadsheet

CRecordset \*m\_rSheet; // Recordset for Excel spreadsheet

};

#endif