DAT: Matrices and raw data editor.

Dietmar G. Schrausser

2023

Overview

Matrices and raw data editor for SCHRAUSSER-MAT (Schrausser, 2022). Application for MS Windows (Schrausser, 2023).

C++ Source of main functions

```
//-------------------------------------------------------------------------| datDlg.cpp
	//
//                                                                        | DAT, Matrix Dialog
	v2.0           (deutsch)  //
//                                                                        | von Dietmar
	Schrausser, (C) SCHRAUSSER 2011  //
//

#include "stdafx.h"
#include "dat.h"
#include "datDlg.h"
#include "dat_index.h"
#include "dat_zellen.h"
#include "dat_aij.h"
#include "D:\_EIGENEDATEIEN_\1_LAUFENDES\1_SYSTEM\3_C_PROGRAMME\__H_C++_\DATACONV.HPP"

#define N_ 10000 // datenmatrix Matrix n
#define K_ 550   // datenmatrix Matrix k


#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

int n_=1, nk_=0, nd_=0, kd_=0;
int sc_ind=0, sc_indx=0, sc_ind_h=0, sc_ind_hx=0, gx, rdw=-1, gii_=1, gix=1, gij_=1, gjx=1;
int sp=80, spx=80, zl=12, zlx=80; //spaltebreite, zeilenabstand
int fhg; //hintergrundfarben

int sw_xs=1, sw_xsx=1;


char datenmatrix[N_][K_][40];


CString n_c;
CString k_c;
CString nk_c;

CString cl_filename;


CdatDlg::CdatDlg(CWnd* pParent)
	: CDialog(CdatDlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CdatDlg)
	//}}AFX_DATA_INIT
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CdatDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
```

```
	//{{AFX_DATA_MAP(CdatDlg)
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CdatDlg, CDialog)
	//{{AFX_MSG_MAP(CdatDlg)
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_WM_SIZE()
	ON_WM_HSCROLL()
	ON_WM_VSCROLL()
	ON_WM_TIMER()
	ON_WM_MOUSEWHEEL()
	ON_COMMAND(ID_DATEI_OEFFNEN, OnDateiOeffnen)
	ON_COMMAND(ID_EINSTELLUNGEN_SCHRIFTART, OnEinstellungenSchriftart)
	ON_COMMAND(ID_EINST_FARBEN, OnEinstFarben)
	ON_WM_CLOSE()
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

void CdatDlg::cmdline(CString f)  // cmdline argument
//                                 // dat.exe [Matrix-Dateiname]
//                                 // zB: dat MAT_F.txt
//                                 // bei falschem argument öffnen von DAT.txt
//                                 // bei keinem argument -> sonst
{
	cl_filename=              f;   // cmdline argument matrix-dateiname
};


BOOL CdatDlg::OnInitDialog()
{
	CDialog::OnInitDialog();

	     CWinApp* pApp = AfxGetApp(); // ini profil lesen

	schr.fn=      pApp->GetProfileString("Schriftart","Name","Lucida Sans Unicode");
	schr.clr=     pApp->GetProfileInt("Schriftart","Farbe",0);
	schr.H=       pApp->GetProfileInt("Schriftart","Höhe",13);
	schr.W=       pApp->GetProfileInt("Schriftart","Breite",4);
	schr.sz=      pApp->GetProfileInt("Schriftart","Grösse",8);

	fhg=          pApp->GetProfileInt("Farbe","Hintergrund",16777215);

	datname=      pApp->GetProfileString("Datenmatrix","Pfad","DAT.txt");
	datname_name= pApp->GetProfileString("Datenmatrix","Name","DAT.txt");

	coord.left=   pApp->GetProfileInt("Dialog","Position_x",579);
	coord.top=    pApp->GetProfileInt("Dialog","Position_y",151);
	coord.right=  pApp->GetProfileInt("Dialog","Grösse_x",399);
	coord.bottom= pApp->GetProfileInt("Dialog","Grösse_y",268);

	SetWindowPos(&wndTop, coord.left,
	                           coord.top,
	                                       coord.right+8,
	                                       coord.bottom+45, SWP_SHOWWINDOW); //hauptfenster
	position
/**/
	if(cl_filename!="")                   //cmdline argument
	{
		datname=cl_filename;
	}

	if (fopen (datname, "r") == 0) //autogenerierte matrixdatei
	{
	          FILE *f;         datname="DAT.txt";
	                f = fopen (datname, "w");

		fprintf( f,"1\t6\n");
		fprintf( f,"2\t3\n");
		fprintf( f,"3\t8\n");
		fprintf( f,"4\t2\n");
		fprintf( f,"5\t6\n");
		fprintf( f,"6\t3\n");
		fprintf( f,"7\t9\n");
		fprintf( f,"8\t4\n");
		fprintf( f,"9\t2\n");
		fprintf( f,"10\t9\n");
```

```
		fclose(  f );
	}

	SetTimer(0,100,0);

	SetIcon(m_hIcon, 1); SetIcon(m_hIcon, 0);

	datname_name=datname;//fenstertext dateiname

	dat_in(); //datenmatrix darstellen

	return 1;
}


void CdatDlg::OnPaint()
{
	UpdateWindow();

	CPaintDC ooo(this);

	CFont font[3];
	      font[1].CreateFont(schr.H,  schr.W,
	0,0,400,0,0,0,0,OUT_DEFAULT_PRECIS,CLIP_DEFAULT_PRECIS,DEFAULT_QUALITY,DEFAULT_PITCH,sc
	hr.fn );
	         font[2].CreateFont(13,      4,
	0,0,400,0,0,0,0,OUT_DEFAULT_PRECIS,CLIP_DEFAULT_PRECIS,DEFAULT_QUALITY,DEFAULT_PITCH,"L
	ucida Sans Unicode" );

	CPen ln[3];
	     ln[1].CreatePen(PS_SOLID, 1,16777215); //linie weiss
	     ln[2].CreatePen(PS_SOLID, 1,8421504 ); //linie grau

	CRect oo1(54, 19, dlg.x, dlg.y-19);ooo.FillSolidRect(oo1, fhg        );    //
	hintergrund
	CRect oo2(0,  0,     54, dlg.y   );ooo.FillSolidRect(oo2,  13357270  );    // balken
	links

	ooo.SelectObject(&ln[2]);ooo.MoveTo( 53, 0);ooo.LineTo( 53, dlg.y);      // linie links
	grau
	ooo.SelectObject(&ln[1]);ooo.MoveTo( 54, 0);ooo.LineTo( 54, dlg.y);      // linie links
	weiss

	CRect oo3(0, 0,  dlg.x,  19      );ooo.FillSolidRect(oo3,  13357270  );    // balken
	oben

	ooo.SelectObject(&ln[2]);ooo.MoveTo( 0, 18);ooo.LineTo( dlg.x, 18);      // linie oben
	grau
	ooo.SelectObject(&ln[1]);ooo.MoveTo( 0, 19);ooo.LineTo( dlg.x, 19);      // linie oben
	weiss

	ooo.SelectObject(&font[2]);
	ooo.SetTextColor(0);
	ooo.SetBkColor( 13357270);

	if(rdw==1)
	{
		if(gij_==1)                                              //index j ausgabe
		{
				ooo.SetBkColor( 13357270);
			ooo.TextOut(67-11,5, "j:");

			for(int ij=0;
				ij<nk_/n_ - sc_ind_h;
					ij++)
			{
					ooo.TextOut(67  +ij * sp , 5  , itoc(ij+1+ sc_ind_h));
			}
		}

		if(gii_==1)                                             //index i ausgabe
		{
			ooo.SetBkColor( 13357270);
			ooo.TextOut(7,5, "i:");

			for(int ii=1;
				ii<=nd_;
```

```
                                        ii++)
                        {
                                        if(ii+sc_ind<=n_)
                                        ooo.TextOut(7          , 9  +ii * zl, itoc(ii+sc_ind));
                        }
                }

                for(int i=1;
                        i<=nd_;
                                i++)                                        //datenausgabe
                for(int j=0;
                        j<kd_;
                                j++)//nk_/n_
                {
                                ooo.SelectObject(&font[1]);
                                ooo.SetTextColor(schr.clr);
                            ooo.SetBkColor( fhg);
                                ooo.TextOut(67 +j * sp , 9  +i * zl, datenmatrix[i+ sc_ind
        ][j+1+ sc_ind_h ]);
                                          //^  ^^ //^^ //^  ^^  //^^
        //^^^^^^      //^^^^^^^^^
                                         //linkerrand //       //                scroll
        Vert     scroll Horz
                                                     //spaltenbreite
                                                            //      //
                                                       //obererrand
                                                                    //
                                                                   //Zeilenabstand
                }
        }

        CRect oo4(0, dlg.y-19, dlg.x, dlg.y);ooo.FillSolidRect(oo4, 13357270);         //
        balken unten

        ooo.SelectObject(&ln[1]);ooo.MoveTo(0,dlg.y-19);ooo.LineTo(dlg.x,dlg.y-19);  // linie
        unten

        if(rdw==1)
        {
                ooo.SelectObject(&font[2]);
                ooo.SetTextColor(0);
                ooo.TextOut(dlg.x-60,5,k_c);             //k ausgabe
                ooo.TextOut(7,dlg.y-15,n_c);             //n ausgabe
                ooo.TextOut(dlg.x-60,dlg.y-15,nk_c);     //nk ausgabe
        }
}

HCURSOR CdatDlg::OnQueryDragIcon(){return (HCURSOR) m_hIcon;}


void CdatDlg::OnSize(UINT nType, int cx, int cy)
{
        CDialog::OnSize(nType, cx, cy);

        dlg.x= cx;
        dlg.y= cy;

        RedrawWindow();
}

void CdatDlg::OnHScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar)
{
        if(nSBCode==SB_LINELEFT)    { if(sc_ind_h > 0)      sc_ind_h--;
        SetScrollPos(SB_HORZ, sc_ind_h, 1); }
        if(nSBCode==SB_THUMBTRACK)  {                       sc_ind_h= nPos;
        SetScrollPos(SB_HORZ, sc_ind_h, 1); }
        if(nSBCode==SB_LINERIGHT)   { if(sc_ind_h < nk_/n_) sc_ind_h++;
        SetScrollPos(SB_HORZ, sc_ind_h, 1); }



        RedrawWindow();

        CDialog::OnHScroll(nSBCode, nPos, pScrollBar);
}

void CdatDlg::OnVScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar)
{
```

```
        if(nSBCode==SB_LINEUP)     { if(sc_ind > 0)  sc_ind--;     SetScrollPos(SB_VERT,
        sc_ind, 1); }
        if(nSBCode==SB_THUMBTRACK) {                 sc_ind= nPos; SetScrollPos(SB_VERT,
        sc_ind, 1); }
        if(nSBCode==SB_LINEDOWN)   { if(sc_ind < n_) sc_ind++;     SetScrollPos(SB_VERT,
        sc_ind, 1); }


        RedrawWindow();

        CDialog::OnVScroll(nSBCode, nPos, pScrollBar);
}


BOOL CdatDlg::OnMouseWheel(UINT nFlags, short zDelta, CPoint pt)
{
        if(                       sc_ind >= 0  &&
                                     sc_ind <= n_    )
        {
                                     sc_ind-=(zDelta/120);
                                  if(sc_ind<= 0 ) sc_ind= 0;
                                  if(sc_ind>= n_) sc_ind= n_;
               SetScrollPos(SB_VERT, sc_ind, 1);

            RedrawWindow();
        }

        return CDialog::OnMouseWheel(nFlags, zDelta, pt);
}

void CdatDlg::OnEinstFarben() //hintergrundfarbe
{
        CColorDialog cl(fhg,0,0);
                     cl.DoModal();
                    fhg=cl.GetColor();

        RedrawWindow();
}


void CdatDlg::OnEinstellungenSchriftart() //schriftart
{
                        strncpy((char*)lf.lfFaceName,    schr.fn,  sizeof lf.lfFaceName);
                                        lf.lfFaceName[ sizeof lf.lfFaceName-1] = 0;


          CFontDialog f(               &lf, CF_EFFECTS | CF_SCREENFONTS ,0,0);
                   f.DoModal();
           schr.fn=f.GetFaceName();
               schr.sz=f.GetSize();
              schr.clr=f.GetColor();
                            f.GetCurrentFont(&lf);
                                                schr.H=lf.lfHeight;
                                                schr.W=lf.lfWidth;

        RedrawWindow();
}

void CdatDlg::OnDateiOeffnen() //datenmatrixdatei öffnen
{
                                 static char BASED_CODE szFilter[]="ASCII Matrix-Dateien (*.asc)
        |*.asc|ASCII Text Matrix-Dateien (*.txt) |*.txt|Alle Dateien (*.*) |*.*||";

         CFileDialog f(1,"asc","*.asc", OFN_HIDEREADONLY |
                                                                  OFN_NOVALIDATE  , szFilter);
                               f.DoModal();
   datname_name= f.GetFileName();
            datname= f.GetPathName();

        if(datname_name!="")dat_in();
}


void CdatDlg::dat_in() //daten enlesen
{
        char v[20];
        n_=0, nk_=-1;
```

```
        if(fopen(                datname, "r")!=0)
        {
                        FILE *p;
                                    p=fopen(datname, "r");
         do { if(fgetc(p)=='\n') n_++; }
             while  (feof (p) == 0);
                            fclose(p);
                                    p=fopen(datname, "r");
         do {   fscanf(p,"%s",v); nk_++; }
             while  (feof (p) == 0);
                            fclose(p);
                for(int i1=1;i1<N_;i1++)for(int j1=1;j1<100;j1++) strcpy(datenmatrix[i1][j1],"
        ");
                                    p=fopen(datname, "r");
                for(int i=1;i<=n_;i++)
                for(int j=1;j<=nk_/n_;j++)
                {
                            fscanf(p,"%s",datenmatrix[i][j]);
                }
                            fclose(p);


                if(n_>62)nd_=62;        //zeilendisplay n
                else     nd_=n_;

                if(nk_/n_>20)kd_=20;    //spaltendisplay k
                else         kd_=nk_/n_;


         SetScrollRange(SB_VERT, 0, n_,     1);
                SetScrollRange(SB_HORZ, 0, nk_/n_, 1);

                rdw=1; //datendarstellung ein
        }

        n_c="n=";    n_c+=itoc(n_);
        k_c="k=";    k_c+=itoc(nk_/n_);
        nk_c="nk="; nk_c+=itoc(nk_);


                  CString c;
                          c= "DAT - [";
                              c+= datname_name;
                                  c+= "]";
        SetWindowText(c);

        UpdateData(0);
        RedrawWindow();
}

void CdatDlg::OnTimer(UINT nIDEvent)
{
    if(gx==1){rdw=-1;/*datendarstellung aus*/ RedrawWindow();  SetWindowText("DAT");
        CdatDlg::schl(-1);/*global gx zu -1*/}

                                                                                    int rd=0;
        if(gix!=gii_){           gix=gii_;                                                rd=1;
        }
        if(gjx!=gij_){           gjx=gij_;                                                rd=1;
        }
        if(spx!=sp){             spx=sp;                                                  rd=1;
        }
        if(zlx!=zl){             zlx=zl;                                                  rd=1;
        }
        if(sc_indx!=sc_ind){     sc_indx=sc_ind;     SetScrollPos(SB_VERT, sc_ind, 1);    rd=1;
        }
        if(sc_ind_hx!=sc_ind_h){ sc_ind_hx=sc_ind_h; SetScrollPos(SB_HORZ, sc_ind_h, 1); rd=1;
        }
        if(sw_xsx!=sw_xs){       sw_xsx=sw_xs;                                            rd=1;
        }

        if(rd)RedrawWindow();
        CDialog::OnTimer(nIDEvent);
}

//schaltungsfunktionen
void CdatDlg::schl(int x){gx=x; } //funktion wird von menüpunkt schliessen aufgerufen
void CdatDlg::ind_i(int x){gii_=x; } //funktion wird von dat_index aufgerufen
```

```
void CdatDlg::ind_j(int x){gij_=x; } //funktion wird von dat_index aufgerufen
int  CdatDlg::chk_i(){int x; if(gii_==1) x=1; if(gii_==-1) x=0;return x;}
int  CdatDlg::chk_j(){int x; if(gij_==1) x=1; if(gij_==-1) x=0;return x;}
void CdatDlg::sp_(int x){sp=x; }
void CdatDlg::zl_(int x){zl=x; }
int  CdatDlg::sp_r(){return sp;}
int  CdatDlg::zl_r(){return zl;}
void CdatDlg::aij_i(int x){sc_ind=x;}
void CdatDlg::aij_j(int x){sc_ind_h=x;}
int  CdatDlg::aij_ir(){return sc_ind;}
int  CdatDlg::aij_jr(){return sc_ind_h;}
CString CdatDlg::aij_x(){return datenmatrix[sc_ind+1][sc_ind_h+1];}

void CdatDlg::aij_xs(CString x)
{
                                    const char* c = x;
        strcpy(datenmatrix[sc_ind+1][sc_ind_h+1], c);

        sw_xs *=-1;
}


void CdatDlg::OnClose()
{
        GetWindowRect(&coord);

        CWinApp*   pApp = AfxGetApp();

                   pApp->WriteProfileString("Schriftart","Name",schr.fn);
                   pApp->WriteProfileInt("Schriftart","Farbe",schr.clr);
                   pApp->WriteProfileInt("Schriftart","Höhe",schr.H);
                   pApp->WriteProfileInt("Schriftart","Breite",schr.W);
                   pApp->WriteProfileInt("Schriftart","Grösse",schr.sz);

                          pApp->WriteProfileInt("Farbe","Hintergrund",fhg);

                          pApp->WriteProfileString("Datenmatrix","Pfad",datname);
                          pApp->WriteProfileString("Datenmatrix","Name",datname_name);

        if(1)pApp->WriteProfileInt("Dialog","Position_x",coord.left);
        if(1)pApp->WriteProfileInt("Dialog","Position_y",coord.top);
        if(1)pApp->WriteProfileInt("Dialog","Grösse_x",dlg.x);
        if(1)pApp->WriteProfileInt("Dialog","Grösse_y",dlg.y);


        CDialog::OnClose();
}
```

References

Schrausser, D. G. (2022). *SCHRAUSSER-MAT: Funktionsindex.* DOI:10.13140/RG.2.2.28314.52164
Schrausser, D. G. (2023). *Schrausser/DAT: Matrices and raw data editor. (v2.0.0)*. Zenodo. DOI:10.5281/zenodo.7651151