# Schritt für Schritt Anleitung für Datenverwaltung unter MFC

## Ziel, Inhalt

• Wir erhalten hier eine Anleitung um ein einfaches MFC Programm zu erstellen, das als Ausgangspunkt für das Projekt Notenverwaltung dienen kann.

Schritt für Schritt Anleitung für Datenverwaltung unter MFC	1
Ziel, Inhalt	1
Einfache Notenverwaltung, Schritt für Schritt	2
Projekt erzeugen	2
Single - Document Interface	2
Neue Klasse CNote	6
Elemente der Klasse CNote	7
Dialog zum Erzeugen von Noten	9
Zugriff auf den Notenwert im Dialog	13
Menupunkt einfügen	13
Menupunkt behandeln	14
Document Klasse ergänzen	14
CObArray in der Document Klasse	15
Zugriff auf die Noten	16
Anpassen der View Klasse	16

# Einfache Notenverwaltung, Schritt für Schritt

# Projekt erzeugen

Wir erzeugen ein Projekt mit dem MFC - Applikationsassistenten. Du hast die Möglichkeit dieses Projekt zu verwenden um daraus später die endgültige Notenverwaltung zu bauen, oder du kannst es als paralleles Projekt dazu betreiben. Ich verwende den zweiten Ansatz, wo ich ein zweites Projekt erzeuge und die gewonnenen Erkenntnisse danach im endgültigen Projekt verwende.

🚆 ATL COM AppWizard 🛛 💁 Win32 Dynamic	c-Link Library Project name:
💐 ATL/WTL AppWizard 🛛 🕺 Win32 Static Li	brary NotenVerwaltungTest
🔐 Cluster Resource Type Wizard	1
🕂 Custom AppWizard	Lo <u>c</u> ation.
💼 Database Project	C:\Documents and Settings\Mar
DevStudio Add-in Wizard	
🕉 Extended Stored Proc Wizard	
🗬 ISAPI Extension Wizard	Create new workspace
🚰 Makefile	C Add to current workspace
🚰 MFC ActiveX ControlWizard	Dependency of:
🖉 MFC AppWizard (dll)	
⊱ MFC AppWizard (exe)	
🕺 New Database Wizard	
🎁 Utility Project	Distance
🛋 Win32 Application	Elaronnis.
Win32 Console Application	IWIN32
•	The second secon

Applikationsassistent

#### Single - Document Interface

Unser Programm soll nur ein Dokument geöffnet halten können. Sollen mehrere Dokumente bearbeitet werden, muss der Benutzer das Programm noch einmal starten. Diese Interface - Art ist moderner und unterstützt das die Idee an einem Dokument zu arbeiten, anstelle vom Arbeiten mit einem Programm.

1FC AppWizard - Step	1 Jocument 1	What type of	application would y	ou like to create?
File Edit Yiew Windo	₩ <u>H</u> elp	<ul> <li>G Single</li> <li>C Multipl</li> <li>C Dialog</li> <li>I Docun</li> </ul>	<u>document</u> e documents based nent∕⊻iew architectr	ure support?
		What Jangua German [I	ge would you like y Germany] (APPWZ[	Dur resources in? DEU.DLL)
	< <u>B</u> ack	<u>N</u> ext>	Einish	Cancel

Die Auswahl der "Single document" Eigenschaft

Bei Schritt 2 und Schritt 3 belassen wir die Grundeinstellungen, "Datenbankunterstützung" nichts anwählen, obwohl, die Schulnoten schlussendlich in eine Datenbank gehören würden. Auch Schritt 3 ist mit den voreingestellten Werten in Ordnung. So dass wir nun zu Schritt 4 gelangen:

Application	What features would you like to include?
le Edit Yiew Yindow Help	✓ Docking toolbar
rint rint Preview	✓ Initial <u>s</u> tatus bar
rint Setup	Printing and print preview
zit	Context-sensitive H <u>e</u> lp
	☑ <u>3</u> D controls
sadu	MAPI (Messaging API)
	Image: Market Sockets
	How do you want your toolbars to look?
Editing Control: Record	Normal
X Check Boz 💿 Radio Button O Radio Button	C Internet Explorer ReBars
	How many files would you like on your recent file list?
	4 📑 Advanced

Hier wählen wir den Knopf für weitere Einstellungen

Folgender Dialog erscheint nun:

	Level and the U
ocument Template Strings	Window Styles
Non-localized strings	
File e <u>x</u> tension:	File type <u>I</u> D:
not	NotenVerwaltungTest.D
Localized strings	Main frame <u>c</u> aption:
Localized strings	Main frame caption:
Localized strings Language: German [Germany]	Main frame <u>c</u> aption: NotenVerwaltung
Localized strings Language: German [Germany] Doc type name:	Main frame <u>c</u> aption: NotenVerwaltung <u>F</u> ilter name:
Localized strings Language: German [Germany] Doc type name: Noten	Main frame <u>c</u> aption: NotenVerwaltung <u>Filter name:</u> Dateityp Noten (*.not)
Localized strings Language: German [Germany] Doc type name: Noten File <u>n</u> ew name (short name):	Main frame <u>c</u> aption: NotenVerwaltung Eilter name: Dateityp Noten (*.not) File type na <u>m</u> e (long name):

Dialog für weitere Einstellungen

4/17

Bei diesem Dialog können wir unser Programm mit einer Dateiendung verbinden. Auch können wir verschiedene weitere Einstellungen vornehmen, wie der Name des Hauptfensters ("NotenVerwaltung").

Nach Schliessen dieses Dialoges sind die Schritte 4 und 5 einfach mit "weiter>" zu bestätigen.

Beim Schritt 6 wählen wir noch als View Typ nicht den voreingestellten Typ CView sondern die Klasse CListView. Von dieser Klasse wird unsere View Klasse abgeleitet. Die CListView beherbergt ein List Control. Das List Control wird vom Betriebssystem verwendet um die rechte Seite im Explorer darzustellen. Es kann als Icon-, Detail- und Listenansicht verwendet werden.

MFC AppWizard - Step 6 of 6	AppWizard creates the following classes for you: CNotenVerwaltungTestView CNotenVerwaltungTestApp CMainFrame CNotenVerwaltungTestDoc		
	Class name: CNotenVerwaltungTe	H <u>e</u> ader file: NotenVerwaltungTes	
	B <u>a</u> se class: CListView	Implementation file: NotenVerwaltungTes	
< <u>B</u> ack	Next > Ei	nish Cancel	

Als Basisklasse verwenden wir die CListView Klasse

#### Neue Klasse CNote

Diese Klasse leiten wir von CObject ab, um das Serialieren der MFC zu verwenden. Die Klasse kann mit dem Klassenassistenten erzeugt werden.

Workspace	<u>×</u>
NotenVerwaltungTest classes     CAboutDlg     CMainFrame     CNotenVerwaltungTestApp     CNotenVerwaltungTestDoc     CNotenVerwaltungTestView     Globals	Set as Acti <u>v</u> e Project          New Class            •₩ New ATL Object         New Form             •₩ Folder             •₩ Add to Source Control             • Docking View
ClassView ResourceView FileView	Hide Properties

Klassenansicht mit Context Menu

Im Assistenten wählen wir als Klassentyp nicht eine MFC Klasse, denn die Klasse CObject steht dann nicht als Basisklasse zur Verfügung. Wähle also "allgemeine Klasse" als Typ und leite von CObject ab:

Class <u>t</u> ype:	Generic Cl	ass		•	OK
Class information					Cancel
Name:	CNote			- 1 -	
– File name:	Note con				
	1		Change		
Race class(es):			<u>Change</u>		
Derived From		As			
CObject		public			

Neue Klasse mit CObject als Basisklasse

Nach dem OK erscheint eine Warnung, die man einfach bestätigen kann.

#### Elemente der Klasse CNote

Mit dem Klassenassistenten kann nun das Datenelement für den Notenwert ergänzt werden.

dd Member	Variable		?
Variable <u>T</u> ype:			OK
double			Canaal
Variable <u>N</u> ame	e:		
_notenWert			
Access		1	
C Public	C Protected	Private	

Datenelement für die Klasse CNote

Ergänze auch einen Konstruktor mit Parametern und das DECLARE\_SERIAL Makro. Dazu gehört dann auch das überschreiben der Serialize Methode und eine Methode asString damit wir einen String erhalten.

Function <u>Type</u>	:		ОК
CString			Consel
Function <u>D</u> ecl	aration:		Lancel
asString() con	st		
Access		]	
Public	C Protected	C Private	

Die Methode asString einfügen

```
class CNote : public CObject
{
public:
    CNote();
    CNote(double notenWert);
    virtual ~CNote();
    void Serialize(CArchive& ar);
private:
    double _notenWert;
    DECLARE_SERIAL(CNote)
};
```

Die .cpp Datei sieht so aus (nicht vergessen IMPLEMENT\_SERIAL):

```
IMPLEMENT_SERIAL(CNote, CObject, 1)
CNote::CNote()
: notenWert(4.0)
{
}
CNote::CNote(double notenWert)
:_notenWert(notenWert)
{
}
CNote::~CNote()
{
}
void CNote::Serialize(CArchive& ar)
{
    if(ar.IsStoring())
    {
        ar << _notenWert;</pre>
    }
    else
    {
        ar >> notenWert;
    }
}
CString CNote::asString() const
{
    CString string;
    // notenwert als float wert mit
    // zwei Kommastellen
    string.Format("%.2f", _notenWert);
    return string;
}
```

TSU

## Dialog zum Erzeugen von Noten

Im Ressourceneditor erzeugen wir einen Dialog:

Workspace	
⊡ <mark>E]</mark> IDD_ABOUTBOX [German (Ger ⊕ <mark>`</mark> El Icon ⊕ <mark>`</mark> El Menu	Open Binary Data
String Table     String Table     Toolbar     Version     Class//iew	Save NotenVerwaltungTest.rc Check Out Insert Insert Copy
	Inser <u>t</u> Dialog Import Export
	Docking View
	- Properties

In der Ressourcenansicht kann ein neuer Dialog erzeugt werden

Mit den Eigenschaften zu den einzelnen Controls wählst du die Bezeichnungen wie IDD\_NEUENOTE und andere Eigenschaften:

A X	
⊡ 🔄 NotenVerwaltungTest res	Neuen Note einfügen
🗄 💼 Accelerator	ска страната странат
🛱 📲 Dialog	E. Careed
🗄 🧰 Icon	
🗄 🦳 Menu	Dialog Properties
🗄 💼 String Table	-🖼 🎖 General Styles More Styles Extended Styles 🕅 🕢
🗄 💼 Toolbar	
🗄 💼 Version	ID: IDD_NEUENOTE <u>Caption</u> : Neuen Note einfügen
	Font name: MS Sans Serif Menu:
	Font size: 8
	Font ⊻Pos: 0 ⊻Pos: 0 Class name:



Füge eine Editbox ein und einen Text, der dem Benutzer helfen soll.

Neuen Note einfügen	<u>×</u>
Hier neue Note zwischen	ОК
Edit	Cancel
Edit Properties	
🗝 🎖 General   Styles   Exte	nded Styles
ID: IDC_EDIT	
	E Help ID
	1 Toble
La Marine Canada	Help ()

Eigenschaften für die Editbox

Hier noch wie man in einem statischen Text einen neue Zeile erzeugt:

ext Pro	perties			
~ ?	General	Styles	Extended Styles	
ID: ID	C_STATIC		✓ Caption: Zv	wischer <mark>\n</mark> 1.0 und 6.0 ei
✓ Visit	ole	Group	📕 <u>H</u> elp ID	
T Disa	abled	Tab stop		

Mehrzeiliger Text mit \n

Durch Doppelklick auf den Dialog will der Klassenassistent einen neue Klasse erzeugen, was wir dankend annehmen:



Dieser Dialog erscheint durch Doppelklick auf den Dialog im Ressourceneditor

<u>m</u> amo.	CNeueNoteDlg	
File name:	NeueNoteDlg.cpp	Lancei
	<u>C</u> hange	
<u>B</u> ase class:	CDialog 🔹	
<u>D</u> ialog ID:	IDD_NEUENOTE	
Automation		
None		
	1	
C Automation		

Der Assistent hilft beim Erzeugen der Klasse

Jetzt gibt es die Möglichkeit in dieser neuen Klasse ein Datenelement für die Editbox einzubauen:

Message Maps Member	Variables Automation ActiveX Events C	lass Info
Project:	Class <u>n</u> ame:	Add Class 🔻
NotenVerwaltungTest	CNeueNoteDlg	Add Variable
C:\\NeueNoteDlg.h, C:\'	NeueNoteDlg.cpp	Delete Variable
	туре менье	Update Column
IDOK		
)escription:		

11/17

Member variable name:	
 _notenWert	
<u>C</u> ategory:	Lancel
Value 💌	
Variable <u>t</u> ype:	
double 💌	
Davasistian	
Description.	

Datenelement als double - Wert hinzufügen

Wichtig ist, dass wir für das IDC\_EDIT nicht ein Control erzeugen (Category) sondern einen Wert mit dem Datentypen double. Jetzt ist es möglich Grenzen für diese Variable zu definieren.

Project:	Class <u>n</u> a	ame:	Add Class 🔻
NotenVerwaltungTest	CNeue	NoteDlg	
:\\NeueNoteDlg.h, C:\\Neue	eNoteDlg.cpp		
Control <u>I</u> Ds:	Туре	Member	<u>D</u> elete Variable
IDC_EDIT	double	_notenWert	Update <u>C</u> olumn
IDOK			D1 140
escription: double with range	e validation		
<u>1</u> inimum Value: 1.0			
tanimum Values Ind			

Unten können nun Grenzen für die Variable angegeben werden

#### Zugriff auf den Notenwert im Dialog

Mit einer einfachen Get-Methode können wir auf den Notenwert zugreifen. Mit dem Klassenassistenten kann diese in die neue Dialog-Klasse eingefügt werden.

```
double CNeueNoteDlg::getNotenWert() const
{
    return _notenWert;
}
```

So sieht der Code in der Dialog-Klasse aus.

# Menupunkt einfügen



Im Ressourceneditor einen Menupunkt einfügen

Diesen Menupunkt bearbeiten wir in der Document - Klasse.

#### Menupunkt behandeln

Durch Rechtsklick auf die Document Klasse kann man eine "Nachrichtenbehandlungsroutine einfügen" und der Assistent erscheint. Wähle rechts unten das IDM\_NEUENOTE.

ew Windows messages/events:	Existing message/event handlers:	OK
JPDATE_COMMAND_UI	COMMAND	Cancel
		Add Handler
		Add and Edit
		Edit E <u>x</u> isting
	Class or object to handle:	
	IDC_EDIT	
	Filter for monorages publishes to observ	
	Not a Window	

Die Behandlungsroutine für den Menupunkt einfügen

#### Document Klasse ergänzen

Der Code in der Document Klasse (hier CNotenVerwaltungTestDoc) sieht so aus:

zuerst der nötige #include

#include "NeueNoteDlg.h"

Und weiter unten:

```
void CNotenVerwaltungTestDoc::OnNeuenote()
{
    CNeueNoteDlg dlg;
    if(IDOK == dlg.DoModal())
    {
        double notenWert = dlg.getNotenWert();
    }
}
```

Es fehlt noch das Erzeugen eines CNoten - Objektes und das speichern in eine Kollektion.

#### CObArray in der Document Klasse

Mit dem Assistenten fügen wir ein Objekt der Klasse CObArray in die Document Klasse ein:

dd Member \	/ariable		?)
Variable <u>T</u> ype:			ОК
CObArray			Canaal
Variable <u>N</u> ame	¢		
_noten			
Access C <u>P</u> ublic	C Protected	Private	

Datenelement \_noten in die Document-Klasse

Ergänze noch folgendes #include:

```
#include "NeueNoteDlg.h"
#include "Note.h"
```

Und der Code in der Document - Klasse, besteht aus Destruktor und der OnNeueNote Methode:

```
CNotenVerwaltungTestDoc::~CNotenVerwaltungTestDoc()
{
    const int count = noten.GetSize();
    for(int i = 0; i < count; ++i)
    {
        CObject* object = noten[i];
        delete object;
    }
}
void CNotenVerwaltungTestDoc::OnNeuenote()
{
    CNeueNoteDlg dlg;
    if(IDOK == dlg.DoModal())
    {
        double notenWert = dlg.getNotenWert();
        CObject* object = new CNote(notenWert);
        _noten.Add(object);
// Dokument als geändert markieren
        SetModifiedFlag();
        // Views benachrichtigen
        UpdateAllViews(0);
                                }
}
```

#### Zugriff auf die Noten

Es fehlt noch eine Möglichkeit auf das Array zuzugreifen:

Function <u>Type</u>	9:		OK
CObArray&			
Function Decl	aration:		Lancel
getNoten()			
Access		1	
Public	C Protected	C Private	

Zugriff auf das Noten Array

Und hier der Code:

```
CObArray& CNotenVerwaltungTestDoc::getNoten()
{
    return _noten;
}
```

#### Anpassen der View Klasse

Jetzt fehlt noch der Code in der View. Überschreibe zuerst die Methode OnUpdate in unserer View Klasse.

Vew Virtual Functions	Existing virtual function overrides	OK
OnDragOver OnDrop OnEndPrintPreview OnFinalRelease OnNotify OnPrepareDC OnPrint OnScroll OnScrollBy OnUpdate PostNcDestroy PreSubclassWindow PreTranslateMessage Serialize WindowProc WinHelp	OnBeginPrinting     OnDraw     OnEndPrinting     OnInitialUpdate     OnPreparePrinting     PreCreateWindow	Cancel Add Handler Add and Edit Edit Existing

Hier die OnUpdate Methode für die View ergänzen

Füge ein #include für die CNote Klasse ein:

#include "Note.h"

Und hier die Methode OnUpdate:

```
void CNotenVerwaltungTestView::OnUpdate(CView* pSender, LPARAM
lHint, CObject* pHint)
{
    CListCtrl& list = GetListCtrl();
    list.DeleteAllItems();
    CNotenVerwaltungTestDoc* doc = GetDocument();
    CObArray& array = doc->getNoten();
    const int count = array.GetSize();
    for(int i = 0; i < count; ++i)
    {
        CObject* object = array[i];
        if(object->IsKindOf(RUNTIME_CLASS(CNote)))
        {
            CNote* note = (CNote*)object;
            CString notenString = note->asString();
            list.InsertItem(0, notenString);
        }
    }
}
```

Viel Spass.