



Anwendung wertet VB-Projektstrukturen aus.

Inhalt

1. Generierung.....	2
2. Form (TreeView).....	3
3. NotePad++	5
4. SQL-Dateien	6
5. Meldung.....	6
6. Visual Studio	7
7. HTML.....	8
8. Text	9
9. DLL mit Verwendungsnachweis	10
10. Parameterdatei	10



1. Generierung

Form (TreeView) ▾

Zur Generierung der Auswertung stehen 3 Alternativen zur Wahl: Form, HTML und Text. Es werden dabei Elemente¹ aus VB-Programmen extrahiert und situativ in einen definierten Ordner² geschrieben.

Private

Damit werden nebst «Public» auch «Private» deklarierte Elemente ausgewertet.

Ausgewertet werden zudem eingebundene Dynamic Link Library's (DLL) ex «Verweise» mit Visual Studio³. Der Verwendungsnachweis ist im Unterordner «dll» als Textfile dokumentiert.

Sind SQL-Statements vorhanden, werden sie im Unterordner «sql» als Textfile dokumentiert.

Form und HTML erhalten zusätzlich den Link zum DOC-Ordner des Projekts und den Programmdateien.

Automatic

Damit wird der festgelegte Startordner² ausgewertet. Andernfalls kann der Benutzer auswählen.

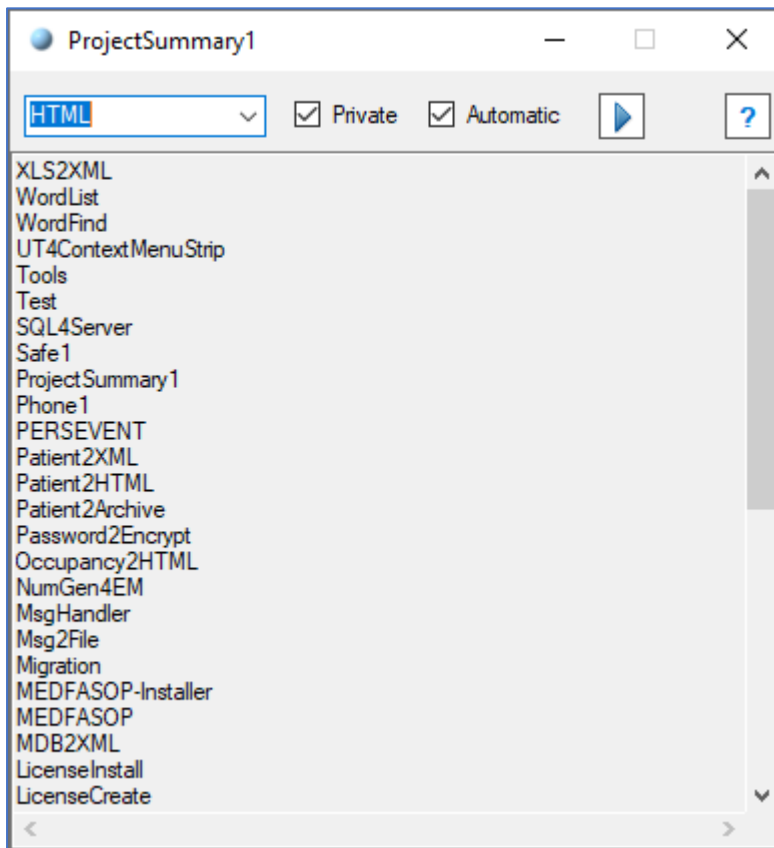





Abbildung 1

Mit  beginnt das Prozedere, welche alle Dateien mit Endung ".vb" im, und unterhalb des Startordners, verarbeitet.

Identifizierte Projekte werden laufend und absteigend aufgelistet.

Mit  wird dieses Dokument im PDF-Format angezeigt.

 Anwendung «WordList» erstellt eine Auflistung der Worte in den VB-Dateien.

¹ Module, Class, Function, Sub, Structure, Enum.

² Gemäss Parameterdatei → S.10.

³ Wenn der Verweis auf den Projekte-DLL-Ordner zeigt.



2. Form (TreeView)

```
ProjectSummary1
├── Adjustment4Patient
│   ├── Type=EXE
│   ├── Integrated DLL=ADO2NET, BTree, LicenseCheck, Password2Encrypt, Tools
│   ├── DOC
│   └── Form1.vb
│       ├── Form1 :: Public Class Form1
│           ├── Ask4Password :: Private Function Ask4Password(tb As String, pwn As String) As String
│           ├── BReplace_Click :: Private Sub BReplace_Click(sender As Object, e As EventArgs) Handles BReplace.Click
│           ├── BStart_Click :: Private Sub BStart_Click(sender As System.Object, e As System.EventArgs) Handles BStart.Click
│           ├── ClearPassword :: Private Function ClearPassword(s As String) As String
│           ├── Extract :: Private Function Extract(ByRef ss As String, fs As String) As String
│           ├── Form1_Load :: Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load
│           ├── GetDate :: Private Function GetDate(cs As String) As String
│           ├── GetUID :: Private Function GetUID() As String
│           ├── Kill4Password :: Private Sub Kill4Password(pwn As String)
│           ├── PDF_Click :: Private Sub PDF_Click(sender As System.Object, e As System.EventArgs) Handles PDF.Click
│           ├── Put4Password :: Private Function Put4Password(tb As String, pwn As String) As Boolean
│           ├── SetDate :: Private Sub SetDate(cs As String, cdt As String)
│           ├── Test4ActiveUser :: Private Function Test4ActiveUser(cs As String) As Boolean
│           ├── Test4Connect :: Private Function Test4Connect(cs As String, pwn As String) As Boolean
│           └── Test4Path :: Private Function Test4Path(ps As String) As Boolean
│       └── Module1.vb
│           ├── Module1 :: Module Module1
│               ├── Action4Row :: Private Function Action4Row(cs As String, sql As String, udt As Date, adt As Date) As myAction
│               ├── AllExOriginal2Copy :: Public Sub AllExOriginal2Copy()
│               ├── cKey [Inherits BTree2.Data] :: Public Class cKey
│                   ├── Compare :: Public Overrides Function Compare(ByRef d As BTree.BTree2.Data) As Short
│               ├── cProtocol [Inherits BTree2.Data] :: Public Class cProtocol
│                   ├── Compare :: Public Overrides Function Compare(ByRef d As BTree.BTree2.Data) As Short
│                   ├── New :: Public Sub New(prole As myRole, ptable As String)
│               ├── cRow [Inherits BTree2.Data] :: Public Class cRow
│                   ├── Compare :: Public Overrides Function Compare(ByRef d As BTree.BTree2.Data) As Short
│                   ├── Documents :: Private Sub Documents(path1 As String, path2 As String, role As myRole, adt As Date)
│                   ├── ExecuteSQL :: Private Sub ExecuteSQL(ByRef sql As String, ByRef cs As String)
│                   ├── GetDTVAL :: Private Function GetDTVAL(ByRef dt As String, ByRef value As String) As String
│                   ├── GetProtocol :: Public Sub GetProtocol()
│                   ├── IsKey :: Private Function IsKey(name As String) As Boolean
│                   ├── LoadKeys :: Private Sub LoadKeys(keys As String)
│                   ├── Msg2Fom :: Sub Msg2Fom(msg As String)
│                   ├── myAction :: Enum myAction As Integer
│                   ├── myRole :: Enum myRole As Integer
│                   ├── NZ :: Public Function NZ(ByVal test As Object, ByVal substitute As Object) As Object
│                   ├── Row2SQL :: Private Function Row2SQL(ByRef ado As adonet, ra As myAction, tab As String) As String
│                   ├── Run4PID :: Private Sub Run4PID(ByVal oDir As IO.DirectoryInfo, p2 As String, role As myRole, adt As Date)
│                   ├── SetProtocol :: Public Sub SetProtocol(role As myRole, table As String, action As myAction)
│                   ├── Started2SQL :: Private Function Started2SQL(sql As String, wsdT As String) As String
│                   ├── Started4SQL :: Private Function Started4SQL(dt As Date) As String
│                   ├── UpdatesExCopy2Original :: Public Sub UpdatesExCopy2Original()
│                   ├── UpdatesExOriginal2Copy :: Public Sub UpdatesExOriginal2Copy()
│                   ├── ZSD :: Private Function ZSD(z As Integer) As String
│                   └── ZSP :: Private Function ZSP(z As Integer) As String
│           └── Version.vb
│               ├── Version :: Module Version
│               └── CheckLicense :: Public Sub CheckLicense()
├── ADO2NET
├── ANLAGE
├── ANLAGE-Installer
├── Anlage1
├── Anlage2
├── Archive2Patient
└── ...
```




Abbildung 2





Projekte bilden die oberste Ebene der Struktur. Die Elemente «Type»⁴ und «Integrated DLL» darunter sind reine Information.

Blau geschriebene Elemente enthalten SQL-Statements. Orange geschrieben sind «Enum» und «Structure».

Formularkopf:

Ein Klick auf  öffnet die ganze Struktur des aktiven Knotens und ein Klick auf  schliesst sie wieder. Ein Klick auf  schliesst sämtliche offenen Knoten. Der Startpfad ist rechtsbündig aufgeführt.

Formularkörper:

Ein Klick auf  öffnet den Knoten um eine Ebene und ein Klick auf  schliesst ihn wieder. Dasselbe bewirkt der Doppelklick auf den Text.

Keyboard in Projektbaum und SQL-Statements:

[Ctrl] auf Zeilen ruft bei «DOC» den Ordner, und auf VB-Dateien sowie unterhalb davon den Editor «Notepad++»⁵ auf, mit der Datei im Lesemodus, wobei die Zeile mit dem Element dort farbig markiert ist (→ S.5).

[Alt] auf Zeilen mit blauer Schrift ruft die SQL-Datei auf (→ Abbildung 3). Ausgewertet sind darin SELECT, UPDATE, INSERT und DELETE ex VB-Datei.⁶

[Enter] auf Zeilen startet Visual Studio mit SLN- und situativ VB-Datei. → S.7, Situation nach [Enter] bei Action4Row.

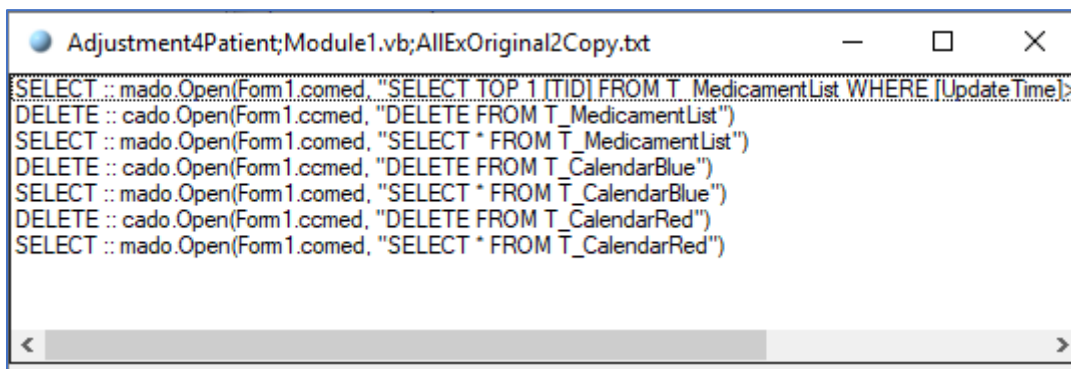


Abbildung 3

⁴ Projekt ist vom Typ «EXE» oder «DLL».

⁵ NotePad++ ist zwingend notwendig.

⁶ [Alt] funktioniert auf Zeilen mit schwarzer oder oranger Schrift analog zu [Ctrl].



3. NotePad++

```
Module1.vb
878 ' in p2 ggf. ordner anlegen
879 If Not IO.Directory.Exists(p2) Then
880     IO.Directory.CreateDirectory(p2)
881 End If
882
883 ' zunächst alle Dateien des Ordners aufspüren
884 For Each oFile In oDir.GetFiles("*.")
885     With oFile
886         SetProtocool(role, "Document", myAction.read)
887         f2 = p2 & "\" & oFile.Name
888         If IO.File.Exists(f2) Then ' check datum
889             If oFile.LastWriteTime.CompareTo(IO.File.GetLastWriteTime(f2)) > 0 AndAlso oFile.LastWriteTime.CompareTo(adt) > 0 Then
890                 IO.File.Copy(oFile.FullName, f2, True)
891                 SetProtocool(f2, "Document", myAction.update)
892             End If
893             ElseIf oFile.CreationTime.CompareTo(adt) < 0 Then
894                 IO.File.Delete(oFile.FullName)
895                 SetProtocool(role, "Document", myAction.delete)
896             Else
897                 IO.File.Copy(oFile.FullName, f2, False)
898                 SetProtocool(f2, "Document", myAction.insert)
899             End If
900         End With
901     Next
902
903 ' Jetzt alle Unterverzeichnis durchlaufen
904 ' und die Prozedur rekursiv selbst aufrufen
905 For Each oSubDir In oDir.GetDirectories()
906     RunSPID(oSubDir, p2 & "\" & oSubDir.Name, role, adt)
907 Next
908 End Sub
909
910 Public Sub AllExOriginal2Copy()
911     Dim msg As String = "Start All ex Original to Copy - "
912     Dim mado As New adonet, cado As New adonet, sql As String = ""
913
914     Msg2Form(msg & "T_MedicamentList")
915     mado.Open(Form1.comed, "SELECT TOP 1 [TID] FROM T_MedicamentList WHERE [UpdateTime]>CONVERT(DATETIME, '' & copydt & '',104)")
916     Dim mlupd As Boolean = mado.HasRows
917     mado.Close()
918     If mlupd Then
919         cado.Open(Form1.ccmcd, "DELETE FROM T_MedicamentList")
920         cado.Close()
921         mado.Open(Form1.comed, "SELECT * FROM T_MedicamentList")
922         LoadKeys("TID")
923         While mado.Read
924             SetProtocool(myRole.original, "T_MedicamentList", myAction.read)
925             sql = Row2SQL(mado, myAction.insert, "T_MedicamentList")
926             ExecuteSQL(sql, Form1.ccmcd)
927             SetProtocool(myRole.copy, "T_MedicamentList", myAction.insert)
928         End While
929         mado.Close()
930     End If
931
932     Msg2Form(msg & "T_CalendarBlue")
933     cado.Open(Form1.ccmcd, "DELETE FROM T_CalendarBlue")
934     cado.Close()
935     mado.Open(Form1.comed, "SELECT * FROM T_CalendarBlue")
936     LoadKeys("SID,From")
937     While mado.Read
938         SetProtocool(myRole.original, "T_CalendarBlue", myAction.read)
939         sql = Row2SQL(mado, myAction.insert, "T_CalendarBlue")
940         ExecuteSQL(sql, Form1.ccmcd)
941         SetProtocool(myRole.copy, "T_CalendarBlue", myAction.insert)
942     End While
943     mado.Close()
944
945     Msg2Form(msg & "T_CalendarRed")
946     cado.Open(Form1.ccmcd, "DELETE FROM T_CalendarRed")
947     cado.Close()
948     mado.Open(Form1.comed, "SELECT * FROM T_CalendarRed")
949     LoadKeys("SID,From,Day,Workbegin")
950     While mado.Read
951         SetProtocool(myRole.original, "T_CalendarRed", myAction.read)
952         sql = Row2SQL(mado, myAction.insert, "T_CalendarRed")
953         ExecuteSQL(sql, Form1.ccmcd)
954         SetProtocool(myRole.copy, "T_CalendarRed", myAction.insert)
955     End While
956     mado.Close()
957
958     Msg2Form("End All ex Original to Copy" & vbCrLf)

```

Abbildung 4

Wichtig: Falls eine VB-Quelldatei nach dem Startzeitpunkt der Generierung geändert hat, erfolgt eine Meldung (→ S.6).



4. SQL-Dateien

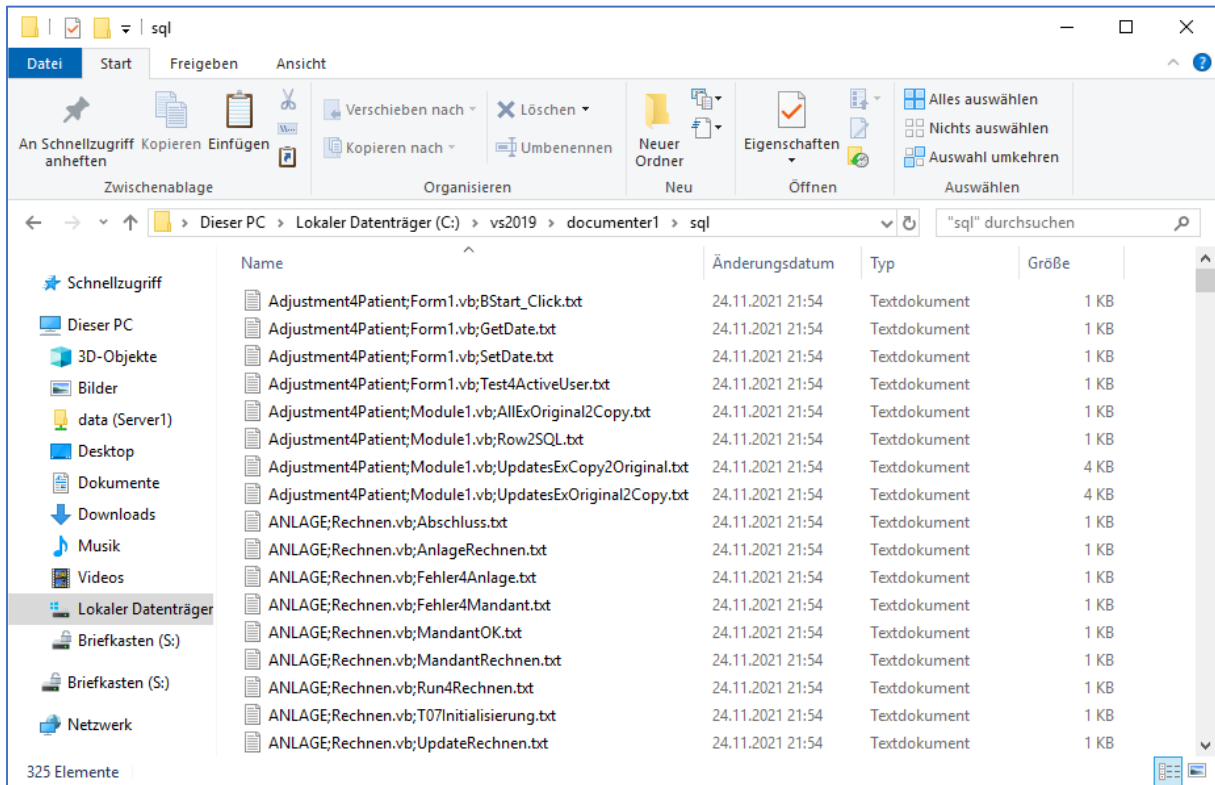


Abbildung 5

Der Dateiname setzt sich aus Projekt-, VB-Datei- und Prozedurname zusammen.

In die Dateien ausgewertet sind SELECT-, UPDATE-, INSERT- und DELETE-Statements, jeweils gefolgt von der Zeilennummer ex VB-Datei.

5. Meldung

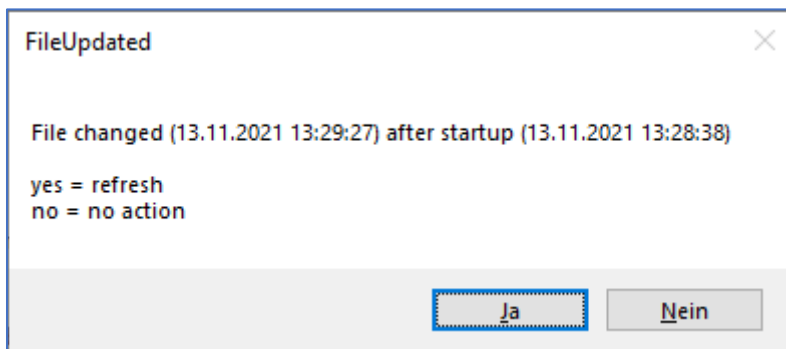


Abbildung 6

Bei Auswahl von «refresh» wird das Projekt aktualisiert, ansonsten wird die Aktion übergangen.



6. Visual Studio

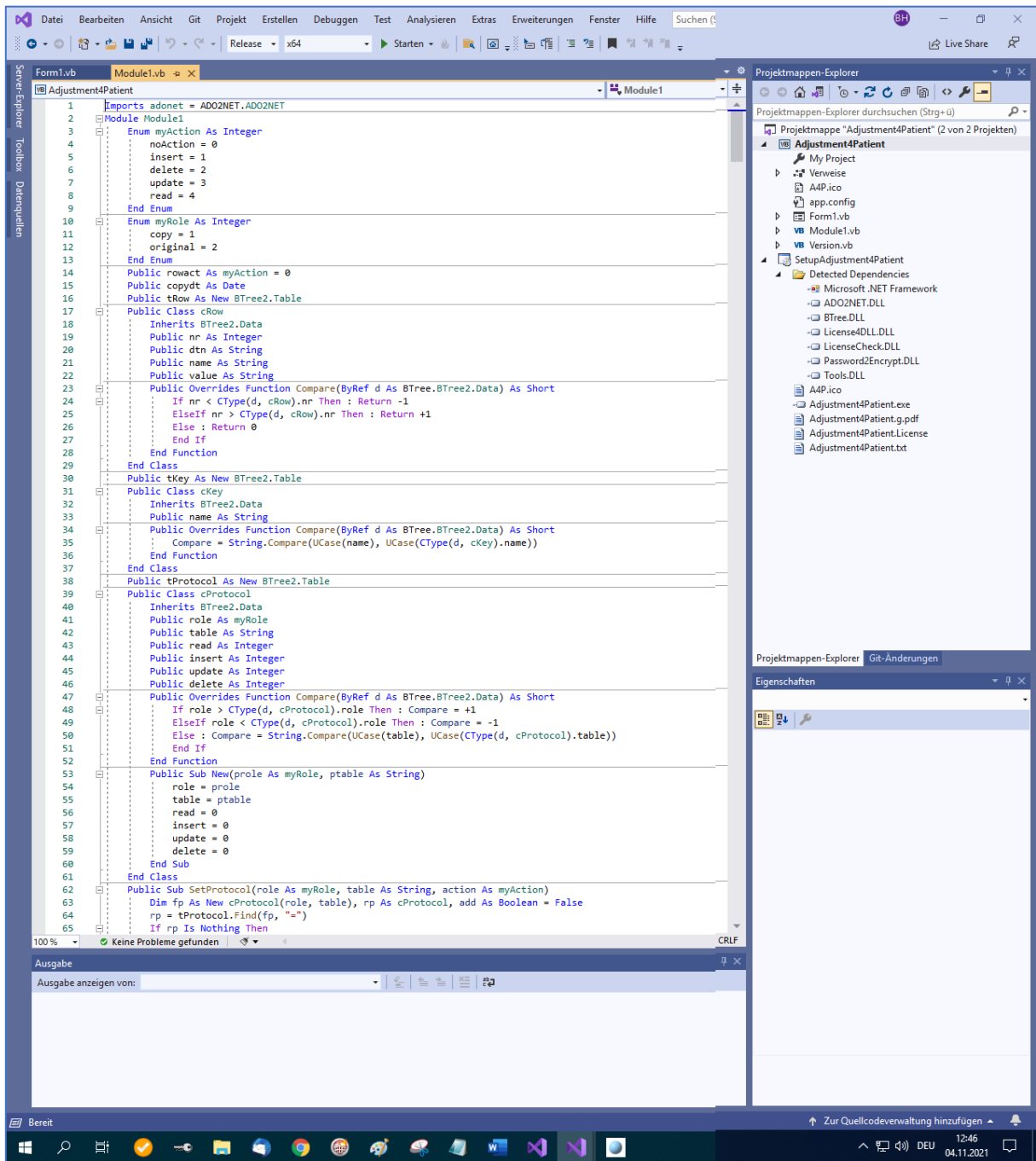


Abbildung 7

Das Bild wurde aus Platzgründen horizontal gestaucht. Visual Studio ist real Programm «devenv.exe»



7. HTML



```
Type=EXE
Integrated DLL=ADO2NET, BTree, LicenseCheck, Password2Encrypt, Tools
DOC=Adjustment4Patient
Form1.vb
| Form1 :: Public Class Form1
| | Ask4Password :: Private Function Ask4Password(tb As String, pwn As String) As String
| | BReplace_Click :: Private Sub BReplace_Click(sender As Object, e As EventArgs) Handles BReplace.Click
| | BStart_Click :: Private Sub BStart_Click(sender As System.Object, e As System.EventArgs) Handles BStart.Click
| | ClearPassword :: Private Function ClearPassword(s As String) As String
| | Extract :: Private Function Extract(ByRef ss As String, fs As String) As String
| | Form1_Load :: Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load
| | GetDate :: Private Function GetDate(cs As String) As String
| | GetUID :: Private Function GetUID() As String
| | Kill4Password :: Private Sub Kill4Password(pwn As String)
| | PDF_Click :: Private Sub PDF_Click(sender As System.Object, e As System.EventArgs) Handles PDF.Click
| | Put4Password :: Private Function Put4Password(tb As String, pwn As String) As Boolean
| | SetDate :: Private Sub SetDate(cs As String, cdt As String)
| | Test4ActiveUser :: Private Function Test4ActiveUser(cs As String) As Boolean
| | Test4Connect :: Private Function Test4Connect(cs As String, pwn As String) As Boolean
| | Test4Path :: Private Function Test4Path(ps As String) As Boolean
Module1.vb
| Module1 :: Module Module1
| | Action4Row :: Private Function Action4Row(cs As String, sql As String, udt As Date, adt As Date) As myAction
| | AllExOriginal2Copy :: Public Sub AllExOriginal2Copy()
| | cKey [Inherits BTree2.Data] :: Public Class cKey
| | | Compare :: Public Overrides Function Compare(ByRef d As BTree.BTree2.Data) As Short
| | cProtocol [Inherits BTree2.Data] :: Public Class cProtocol
| | | Compare :: Public Overrides Function Compare(ByRef d As BTree.BTree2.Data) As Short
| | | New :: Public Sub New(prole As myRole, ptable As String)
| | cRow [Inherits BTree2.Data] :: Public Class cRow
| | | Compare :: Public Overrides Function Compare(ByRef d As BTree.BTree2.Data) As Short
| | Documents :: Private Sub Documents(path1 As String, path2 As String, role As myRole, adt As Date)
| | ExecuteSQL :: Private Sub ExecuteSQL(ByRef sql As String, ByRef cs As String)
| | GetDTVAL :: Private Function GetDTVAL(ByRef dt As String, ByRef value As String) As String
| | GetProtocol :: Public Sub GetProtocol()
| | IsKey :: Private Function IsKey(name As String) As Boolean
| | LoadKeys :: Private Sub LoadKeys(keys As String)
| | Msg2Form :: Sub Msg2Form(msg As String)
| | myAction :: Enum myAction As Integer
| | myRole :: Enum myRole As Integer
```

Abbildung 8

PDF-Dokumente in DOC werden hier direkt angezeigt, andere nur indirekt über den Download. VB-Dateien werden direkt im Lesemodus angezeigt.



8. Text

```
Adjustment4Patient.txt - Editor
Datei Bearbeiten Format Ansicht Hilfe
|Type=EXE
Integrated DLL=AD02NET, BTree, LicenseCheck, Password2Encrypt, Tools

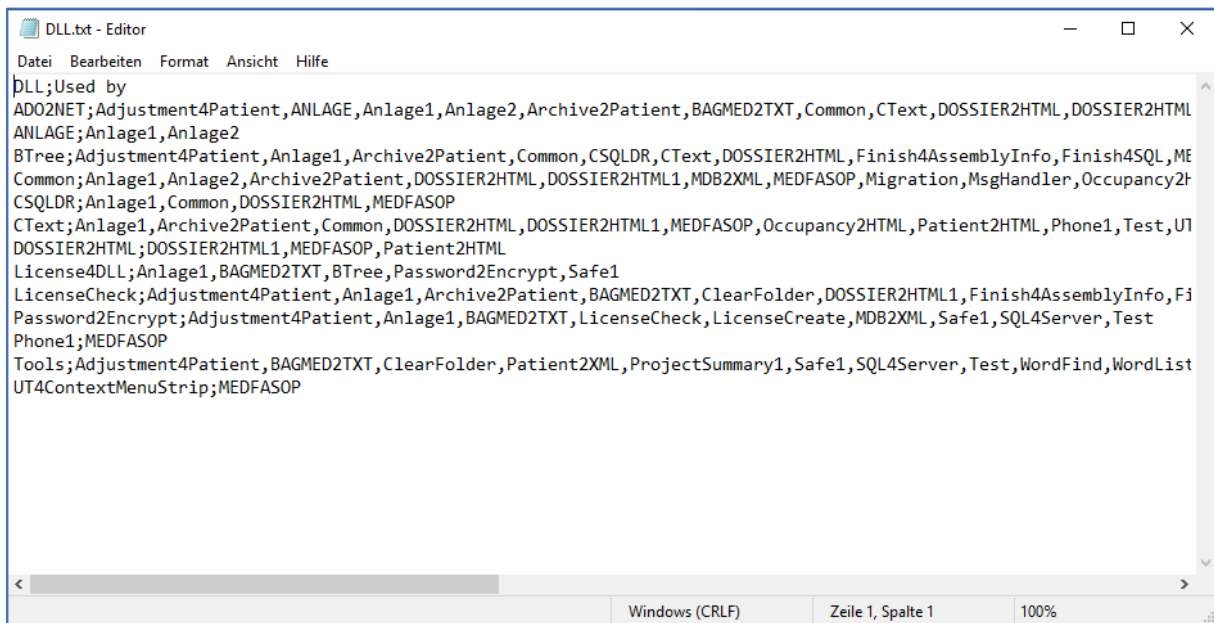
Form1.vb :: Adjustment4Patient\Adjustment4Patient\Form1.vb
| Form1 :: Public Class Form1
| | Ask4Password :: Private Function Ask4Password(tb As String, pwn As String) As String
| | BReplace_Click :: Private Sub BReplace_Click(sender As Object, e As EventArgs) Handles BReplace.Click
| | BStart_Click :: Private Sub BStart_Click(sender As System.Object, e As System.EventArgs) Handles BStart.Click
| | ClearPassword :: Private Function ClearPassword(s As String) As String
| | Extract :: Private Function Extract(ByRef ss As String, fs As String) As String
| | Form1_Load :: Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load
| | GetDate :: Private Function GetDate(cs As String) As String
| | GetUID :: Private Function GetUID() As String
| | Kill4Password :: Private Sub Kill4Password(pwn As String)
| | PDF_Click :: Private Sub PDF_Click(sender As System.Object, e As System.EventArgs) Handles PDF.Click
| | Put4Password :: Private Function Put4Password(tb As String, pwn As String) As Boolean
| | SetDate :: Private Sub SetDate(cs As String, cdt As String)
| | Test4ActiveUser :: Private Function Test4ActiveUser(cs As String) As Boolean
| | Test4Connect :: Private Function Test4Connect(cs As String, pwn As String) As Boolean
| | Test4Path :: Private Function Test4Path(ps As String) As Boolean
Module1.vb :: Adjustment4Patient\Adjustment4Patient\Module1.vb
| Module1 :: Module Module1
| | Action4Row :: Private Function Action4Row(cs As String, sql As String, udt As Date, adt As Date) As myAction
| | AllExOriginal2Copy :: Public Sub AllExOriginal2Copy()
| | cKey [Inherits BTree2.Data] :: Public Class cKey
| | | Compare :: Public Overrides Function Compare(ByRef d As BTree.BTree2.Data) As Short
| | cProtocol [Inherits BTree2.Data] :: Public Class cProtocol
| | | Compare :: Public Overrides Function Compare(ByRef d As BTree.BTree2.Data) As Short
| | | New :: Public Sub New(prole As myRole, ptable As String)
| | cRow [Inherits BTree2.Data] :: Public Class cRow
| | | Compare :: Public Overrides Function Compare(ByRef d As BTree.BTree2.Data) As Short
| | Documents :: Private Sub Documents(path1 As String, path2 As String, role As myRole, adt As Date)
| | ExecuteSQL :: Private Sub ExecuteSQL(ByRef sql As String, ByRef cs As String)
| | GetDTVAL :: Private Function GetDTVAL(ByRef dt As String, ByRef value As String) As String
| | GetProtocol :: Public Sub GetProtocol()
| | IsKey :: Private Function IsKey(name As String) As Boolean
| | LoadKeys :: Private Sub LoadKeys(keys As String)
| | Msg2Form :: Sub Msg2Form(msg As String)
| | myAction :: Enum myAction As Integer
| | myRole :: Enum myRole As Integer
| | NZ :: Public Function NZ(ByVal test As Object, ByVal substitute As Object) As Object
| | Row2SQL :: Private Function Row2SQL(ByRef ado As adonet, ra As myAction, tab As String) As String
| | Run4PID :: Private Sub Run4PID(ByVal oDir As IO.DirectoryInfo, p2 As String, role As myRole, adt As Date)
| | SetProtocol :: Public Sub SetProtocol(role As myRole, table As String, action As myAction)
| | Started2SQL :: Private Function Started2SQL(sql As String, wsdt As String) As String
| | Started4SQL :: Private Function Started4SQL(dt As Date) As String
| | UpdatesExCopy2Original :: Public Sub UpdatesExCopy2Original()
| | UpdatesExOriginal2Copy :: Public Sub UpdatesExOriginal2Copy()
| | ZSD :: Private Function ZSD(z As Integer) As String
| | ZSP :: Private Function ZSP(z As Integer) As String
Version.vb :: Adjustment4Patient\Adjustment4Patient\Version.vb
| Version :: Module Version
| | CheckLicense :: Public Sub CheckLicense()

Windows (CRLF) Zeile 1, Spalte 1 100%
```

Abbildung 9



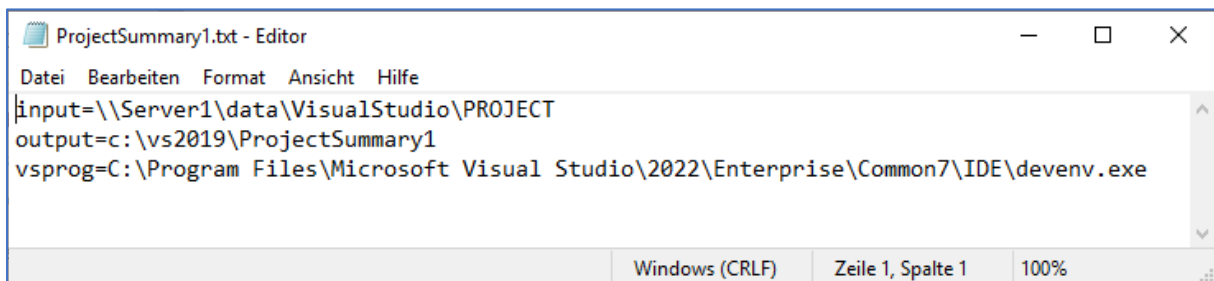
9. DLL mit Verwendungsnachweis



```
 DLL;Used by
ADO2NET;Adjustment4Patient,ANLAGE,Anlage1,Anlage2,Archive2Patient,BAGMED2TXT,Common,CText,DOSSIER2HTML,DOSSIER2HTML
ANLAGE;Anlage1,Anlage2
BTree;Adjustment4Patient,Anlage1,Archive2Patient,Common,CSQLDR,CText,DOSSIER2HTML,Finish4AssemblyInfo,Finish4SQL,ME
Common;Anlage1,Anlage2,Archive2Patient,DOSSIER2HTML,DOSSIER2HTML1,MDB2XML,MEDFASOP,Migration,MsgHandler,Occupancy2F
CSQLDR;Anlage1,Common,DOSSIER2HTML,MEDFASOP
CText;Anlage1,Archive2Patient,Common,DOSSIER2HTML,DOSSIER2HTML1,MEDFASOP,Occupancy2HTML,Patient2HTML,Phone1,Test,UI
DOSSIER2HTML;DOSSIER2HTML1,MEDFASOP,Patient2HTML
License4DLL;Anlage1,BAGMED2TXT,BTree>Password2Encrypt,Safe1
LicenseCheck;Adjustment4Patient,Anlage1,Archive2Patient,BAGMED2TXT,ClearFolder,DOSSIER2HTML1,Finish4AssemblyInfo,Fi
Password2Encrypt;Adjustment4Patient,Anlage1,BAGMED2TXT,LicenseCheck,LicenseCreate,MDB2XML,Safe1,SQL4Server,Test
Phone1;MEDFASOP
Tools;Adjustment4Patient,BAGMED2TXT,ClearFolder,Patient2XML,ProjectSummary1,Safe1,SQL4Server,Test,WordFind,WordList
UT4ContextMenuStrip;MEDFASOP
```

Abbildung 10

10. Parameterdatei



```
input=\\Server1\data\VisualStudio\PROJECT
output=c:\vs2019\ProjectSummary1
vsprog=C:\Program Files\Microsoft Visual Studio\2022\Enterprise\Common7\IDE\devenv.exe
```

Abbildung 11

Die Datei "ProjectSummary1.txt" ist im Programmordner der Anwendung abgelegt und beinhaltet die 3 Parameter für Input, Output und VisualStudio-Programm.