```
Imports System.IO

Public Class Form1
    Enum GDTTyp
        Ohne
        Neu
        Bestand
    End Enum
    Public CFGfile As String = "\PDFEncrypter.cfg"
    Public Loglevel As Integer = 0
    Public LoglevelString As String = "Loglevel="
    Public GDTImportFile As String = ""
    Public GDTImportFileString As String = "GDTImport="
    Public BDTImportFile As String = ""
    Public BDTImportFileString As String = "BDTImport="
    Public xDTSchluesselfeld As String = ""
    Public xDTSchluesselfeldString As String = "xDTSchluesselfeld="
    Public Email As Boolean = False
    Public EmailString As String = "Email="
    Public ExportPath As String = ""
    Public ExportPathString As String = "ExportPath="
    Public ExportPrefix As String = ""
    Public ExportPrefixString As String = "ExportPrefix="
    Public Encrypter As String = ""
    Public EncrypterString As String = "Encrypter="
    Public EmailExe As String = ""
    Public EmailExeString As String = "EmailExe="
    Public EmailFrom As String = ""
    Public EmailFromString As String = "EmailFrom="

    Dim GDTINSatzart As String = ""
    Dim GDTINSatzlaenge As String = ""
    Dim GDTINGDTVersion As String = ""
    Dim GDTINGebuehrenordnung As String = ""
    Dim GDTINPatientennummer As String = ""
    Dim GDTINPatientName As String = ""
    Dim GDTINPatientVorname As String = ""
    Dim GDTINPatientGeburt As String = ""
    Dim GDTINPatientWohnort As String = ""
    Dim GDTINPatientStrasse As String = ""
    Dim GDTINPatientGeschlecht As String = ""
    Dim GDTINPatientVKNR As String = ""
    Dim GDTINPatientGO As String = ""
    Dim GDTINPatientGroesse As String = ""
    Dim GDTINPatientGewicht As String = ""
    Dim GDTINVerfahrensspezifischesKennfeld As String = ""
    Dim GDTINUntersuchungsDatum As String = ""
    Dim GDTINUntersuchungsZeit As String = ""
    Dim GDTINEmail As String = ""
    Dim GDTINTermin1 As String = ""
    Dim GDTINTermin2 As String = ""
    Dim GDTINTermin3 As String = ""


    Dim GDTModus As GDTTyp = GDTTyp.Ohne
    Dim Floskel As String
    Dim IstFehlerfrei As Boolean = True
    Dim Anrede As String
    Public Vorname As String
    Public Nachname As String
    Public Termine As String
    Public TermineKurz As String
    Dim StatusIstOK As Boolean = False
    Dim StatusText As String = "0"
    Dim Nummern As New List(Of String)
    Dim OriginalNummern As New List(Of String)
    Dim Vorwahlen() As String = {"015", "016", "017"}

    Dim Quelle As String = ""
```

```
70      Dim Ziel As String = ""
71      Dim Schluessel As String = ""
72
73      Private Sub Log(ByVal Level As Int16, ByVal Text As String)
74          If Level < Loglevel + 1 Then
75              Dim ExePfad As String = Path.GetDirectoryName(Application.ExecutablePath)
76              Dim Logfile As String = ExePfad + "\Logfile.txt"
77              Dim Zeit As String = DateTime.Now.ToString("yyyy-MM-dd HH:mm:ss:fff")
78              Dim txt As String = Zeit + "  " + Text + vbCrLf
79              File.AppendAllText(Logfile, txt)
80          End If
81      End Sub
82      Private Sub ReadCFG()
83          Dim ExePfad As String = Path.GetDirectoryName(Application.ExecutablePath)
84          CFGfile = ExePfad + CFGfile
85          If File.Exists(CFGfile) Then
86              Dim ReadText() As String = File.ReadAllLines(CFGfile)
87              Dim s As String
88              For Each s In ReadText
89                  If s.StartsWith(LoglevelString) Then
90                      Log(1, "CFG-File: " + s)
91                      Loglevel = CInt(s.Substring(LoglevelString.Length))
92                  End If
93                  If s.StartsWith(GDTImportFileString) Then
94                      Log(5, "CFG-File: " + s)
95                      GDTImportFile = s.Substring(GDTImportFileString.Length)
96                  End If
97                  If s.StartsWith(BDTImportFileString) Then
98                      Log(5, "CFG-File: " + s)
99                      BDTImportFile = s.Substring(BDTImportFileString.Length)
100                 End If
101                 If s.StartsWith(xDTSchluesselfeldString) Then
102                     Log(5, "CFG-File: " + s)
103                     xDTSchluesselfeld = s.Substring(xDTSchluesselfeldString.Length)
104                 End If
105                 If s.StartsWith(ExportPathString) Then
106                     Log(10, "CFG-File: " + s)
107                     ExportPath = s.Substring(ExportPathString.Length)
108                 End If
109                 If s.StartsWith(ExportPrefixString) Then
110                     Log(10, "CFG-File: " + s)
111                     ExportPrefix = s.Substring(ExportPrefixString.Length)
112                 End If
113                 If s.StartsWith(EncrypterString) Then
114                     Log(10, "CFG-File: " + s)
115                     Encrypter = s.Substring(EncrypterString.Length)
116                 End If
117                 If s.StartsWith(EmailExeString) Then
118                     Log(10, "CFG-File: " + s)
119                     EmailExe = s.Substring(EmailExeString.Length)
120                 End If
121                 If s.StartsWith(EmailString) Then
122                     Log(10, "CFG-File: " + s)
123                     Email = CBool(s.Substring(EmailString.Length))
124                 End If
125                 If s.StartsWith(EmailFromString) Then
126                     Log(10, "CFG-File: " + s)
127                     EmailFrom = s.Substring(EmailFromString.Length)
128                 End If
129             Next
130         Else
131         End If
132         Log(3, "CFG-File: " + CFGfile)
133     End Sub
134     Private Sub ReadxDT(ByVal Datei As String)
135         Log(10, "ReadxDT: " + Datei)
136         If Not File.Exists(Datei) Then
137             Log(3, "ReadxDT: Nicht gefunden: " + Datei)
138             ' MessageBox.Show("keine Datei da: " + Datei)
```

```
            Return
        End If
        Dim ReadText() As String = File.ReadAllLines(Datei, System.Text.Encoding.UTF8)
        Dim s As String = ""
        Dim temp As String = ""
        Dim Zeileninhalt As String = ""
        Dim Zeilentyp As String = ""
        Dim Ziffernfolge As String = ""
        For Each s In ReadText
            Zeilentyp = s.Substring(3, 4)
            Zeileninhalt = s.Substring(7)
            Log(10, "ReadxDT: " + Datei + "  " + Zeilentyp + ": " + Zeileninhalt)
            If Zeilentyp = xDTSchluesselfeld Then
                Schluessel = Zeileninhalt
                Log(5, "ReadxDT: Schluessel" + "  " + Zeilentyp + ": " + Zeileninhalt)
            End If

            Try
                Select Case Zeilentyp
                    Case "3000"
                        GDTINPatientennummer = Zeileninhalt
                    Case "3101"
                        GDTINPatientName = Zeileninhalt
                        Nachname = Zeileninhalt
                    Case "3102"
                        GDTINPatientVorname = Zeileninhalt
                        Vorname = Zeileninhalt
                    Case "3103"
                        GDTINPatientGeburt = Zeileninhalt
                    Case "3106"
                        GDTINPatientWohnort = Zeileninhalt
                    Case "3107"
                        GDTINPatientStrasse = Zeileninhalt
                    Case "3110" '1=Mann, 2=Frau
                        If Zeileninhalt = "1" Then
                            Anrede = "Herr"
                            Floskel = "Sehr geehrter Herr"
                        Else
                            Anrede = "Frau"
                            Floskel = "Sehr geehrte Frau"
                        End If
                        GDTINPatientGeschlecht = Zeileninhalt

                    Case "3150", "3151", "3618", "3626"          ' Das sind Telefonnummern.
                    Alle fremden Zeichen ausfiltern!
                        OriginalNummern.Add(Zeileninhalt)
                        OriginalNummern = OriginalNummern.Distinct().ToList()
                        Ziffernfolge = ""
                        For Each t As Char In Zeileninhalt
                            If Char.IsDigit(t) Then Ziffernfolge += t
                        Next
                        If Vorwahlen.Any(Function(b) Ziffernfolge.StartsWith(b)) Then
                            Nummern.Add(Ziffernfolge)
                            Nummern = Nummern.Distinct().ToList()
                        End If
                        Ziffernfolge = ""
                    Case "3619"
                        GDTINEmail = Zeileninhalt
                    Case "3622"
                        GDTINPatientGroesse = Zeileninhalt
                    Case "3623"
                        GDTINPatientGewicht = Zeileninhalt
                    Case "4104"
                        GDTINPatientVKNR = Zeileninhalt
                    Case "4121"
                        GDTINPatientGO = Zeileninhalt
                    Case "6200"
                        GDTINUntersuchungsDatum = Zeileninhalt
                    Case "6201"
```

```
                        GDTINUntersuchungsZeit = Zeileninhalt
                    Case "8000"
                        GDTINSatzart = Zeileninhalt
                    Case "8100"
                        GDTINSatzlaenge = Zeileninhalt
                    Case "8402"
                        GDTINVerfahrensspezifischesKennfeld = Zeileninhalt
                    Case "8403"
                        GDTINGebuehrenordnung = Zeileninhalt
                    Case "9218"
                        GDTINGDTVersion = Zeileninhalt
                End Select
            Catch ex As Exception
                StatusIstOK = False
                StatusText = "Uups. Ein Fehler bei der Datei " + Datei
                Log(1, "ReadxDT: Catch bei " + Datei)
            End Try
        Next

    End Sub
    Private Sub ReadBDT()
        ReadxDT(BDTImportFile)
    End Sub
    Private Sub ReadGDT()
        ReadxDT(GDTImportFile)
    End Sub
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        ReadCFG()
        If GDTImportFile.Length > 0 Then ReadGDT()
        If BDTImportFile.Length > 0 Then ReadBDT()
        Dim args() As String = Environment.GetCommandLineArgs
        For Each arg As String In args
            Log(5, arg)
            If arg.StartsWith("-Quelle:") Then Quelle = arg.Substring("-Quelle:".Length)
            If arg.StartsWith("-Ziel:") Then Ziel = arg.Substring("-Ziel:".Length)
            If arg.StartsWith("-Schluessel:") Then Schluessel = arg.Substring("-Schluessel:".
            Length)
        Next
    End Sub
    Private Sub Form1_Shown(sender As Object, e As EventArgs) Handles MyBase.Shown
        If Ziel.Length < 1 Then
            Ziel = Quelle.Substring(Quelle.LastIndexOf("\") + 1)
        End If
        If Schluessel.Length < 1 Then
            Schluessel = InputBox("Schlüssel", "Schlüssel")
        End If

        Dim args As String = " -applyProfile -profile user/encrypt -profileParam pdf-userPass
        " +
                             Schluessel +
                             " -outputFile " + Chr(34) + ExportPath + ExportPrefix + Ziel +
                              Chr(34) +
                             " " + Chr(34) + Quelle + Chr(34)
        Log(5, "pdf24 Exe:  '" + Encrypter + "'")
        Log(1, "pdf24 args: '" + args + "'")
        If Not File.Exists(Encrypter) Then
            MessageBox.Show("Doc-Tool nicht gefunden!")
        Else
            Dim proc As New Process
            proc = Process.Start(Encrypter, args)
            proc.WaitForExit()
        End If

        If Email Then
            Dim str As String
            str = "-compose " + Chr(34) +
                  "from=" + EmailFrom + "," +
                  "to=" + GDTINEmail + "," +
                  "subject='Ein Dokument für " + GDTINPatientVorname + " " + GDTINPatientName
```

```
                    + "'," +
                    "body=" + Floskel + " " + GDTINPatientName + "," +
                    "attachment=" + ExportPath + ExportPrefix + Ziel +
                    Chr(34)

            Log(5, "TB Exe:  '" + EmailExe + "'")
            Log(1, "TB args: '" + str + "'")
            If Not File.Exists(EmailExe) Then
                MessageBox.Show("Thunderbird nicht gefunden!")
            Else
                Dim TBProc As New Process
                TBProc = Process.Start(EmailExe, str)
            End If
        End If
        Log(5, "Bin fertig")
        Me.Close()
    End Sub
End Class
```