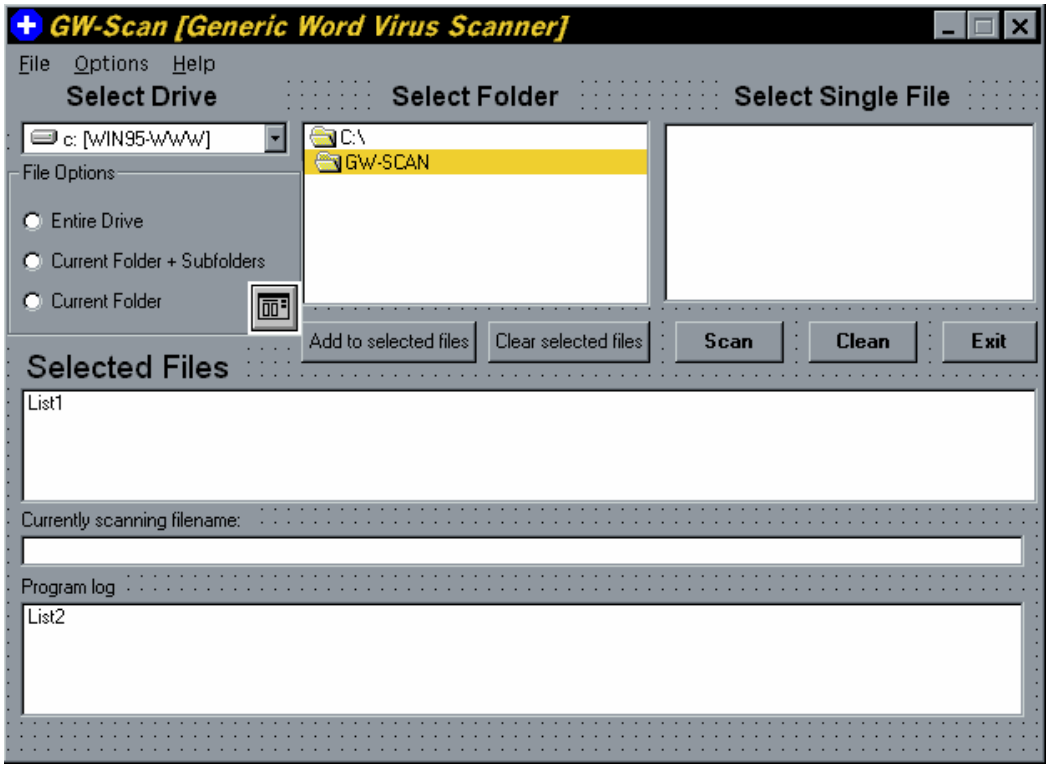
Appendix E - [The Remedy] - Code listing of GW-Scan

The Filebox Form:




```
Dim Virusfound As Integer
Dim Filename, INIFile
Dim Filebuffer  As String 'Global variable to store entire document file contents later
Dim Instpath As String
Private Sub ScanDirs(Toplevel As String, rListBox As ListBox)
'Recursively scan subdirs.
Dim strFiles() As String
Dim iCtr As Integer

On Error GoTo trapper

If Right$(Toplevel, 1) <> "\" Then Toplevel = Toplevel & "\"
ReDim strFiles(0 To 1)

strFiles(0) = Dir$(Toplevel & "*.*", vbDirectory)

Do Until Len(strFiles(0)) = 0
        If strFiles(0) <> String$(Len(strFiles(0)), ".") Then
                strFiles(UBound(strFiles)) = IIf((GetAttr(Toplevel & strFiles(0)) And vbDirectory),
                "*", " ") & strFiles(0)
                ReDim Preserve strFiles(UBound(strFiles) + 1)
        End If
   strFiles(0) = Dir$
Loop

For iCtr = 1 To UBound(strFiles) - 1
If (UCase$(Right$(strFiles(iCtr), 4)) = ".DOT" Or UCase$(Right$(strFiles(iCtr), 4)) = ".DOC") Then
rListBox.AddItem Toplevel & Right$(strFiles(iCtr), Len(strFiles(iCtr)) - 1)
```

```
If Left$(strFiles(iCtr), 1) = "*" Then
   Call ScanDirs(Toplevel & Right$(strFiles(iCtr), Len(strFiles(iCtr)) - 1), rListBox)
End If
Next iCtr

trapper:
   Exit Sub

End Sub
Private Sub Command1_Click()

   ' Clear selected files and reset scanner to initial state

   Filebox.List1.Clear        ' Clear contents of selected files listbox
   Filebox.List2.Clear        ' Clear contents of comments listbox
   Filebox.List3.Clear        ' Clear contents of infected files listbox
   List1.Visible = True       ' Show the selected files box again
   List2.Visible = True       ' Show the comments listbox again
   List3.Visible = False      ' Hide the infected files box from view
   Label1.BackColor = &HC0C0C0 ' Change selected files colour back to normal state
   Label1 = "Selected Files"   ' Changes text from infected files to selected files
   Clean.Enabled = False      ' Disables the clean button until scan finds virus
   Scan.Enabled = False       ' Disabled the scan button until a filelist is created
   Text2.Text = ""            ' Resets the filename being scanned output
   Virusfound = 0
End Sub
Private Sub Command2_Click()
   Unload Me
   End                        ' Terminate the program from the exit button
End Sub
Private Sub Command3_Click()
On Error GoTo trapper
' Set up the listbox variable
Dim MyListBox As ListBox
Set MyListBox = Filebox.List1
' Determine which file option is being used
' and decide if drives, directories or files are to be selected
' Select every document & template from the entire drive
   If Fileopt1.Value = True Then
      Waitbox.Show
      Waitbox.Refresh
      Call ScanDirs(Left$(Drive1.Drive, 2), MyListBox)
      If List2.ListCount > -1 Then Scan.Enabled = True
      Waitbox.Hide
   End If
'Select documents and templates from a directory and it's subdirectories
   If Fileopt2.Value = True Then
      Waitbox.Show
      Waitbox.Refresh
      Call ScanDirs(Dir1, MyListBox)
      'If any files are selected, enable the scan button to be used
      If List1.ListCount > 0 Then Scan.Enabled = True
      Waitbox.Hide
   End If
'Select documents and templates from the current directory only
If Fileopt3.Value = True Then
Dim Pattern As String
Dim FoundFile As String
      For i = 1 To 2
         If i = 1 Then Pattern = "*.DOC" Else Pattern = "*.DOT"
```

```
            ChDrive (Drive1)
            ChDir (Dir1)
            FoundFile = Dir$(Pattern)
            Do Until FoundFile = ""
               If Right$(Dir1, 1) <> "\" Then
                  MyListBox.AddItem Dir1 & "\" & FoundFile
               Else
                  MyListBox.AddItem Dir1 + FoundFile
               End If
               FoundFile = Dir$
            Loop
         Next i

         'If any files are selected, enable the scan button to be used
         If List1.ListCount > -1 Then Scan.Enabled = True
      End If
trapper:
   Exit Sub
End Sub
Private Sub Clean_Click()
   ' The virus cleaner procedure
   ' If the file is not in the exclude list it will:
   ' Reads the input from listbox3 to open a file, seek into it and
   ' Rename the character specified at the byte location in list 3
   ' Below the filename specified in the listbox3
   On Error GoTo trapper
   i = 0
   Do While i < List3.ListCount - 1
      Include = 1
      Open "C:\GW-Scan\EXCLUDE.TXT" For Input As #1
      Do Until EOF(1)
         Line Input #1, A$
         If UCase(A$) = UCase(List3.List(i)) Then Include = 0
      Loop
      Close #1
      If Include = 1 Then
         Open List3.List(i) For Binary Access Write As #1
         Seek #1, List3.List(i + 1)     ' Jump to infected byte location
         Put #1, , "x"                  ' replace first char of macroname with x
         Close #1
      End If
      i = i + 2                         ' Goes to the next filename in the infected list
   Loop
      Call Command1_Click   ' Reset the scanner, it's cleaned all documents in the list
trapper:
   Exit Sub
End Sub
Private Sub Dir1_Change()
   On Error GoTo trapper
   ' Update file list when directory changes
   File1.Path = Dir1.Path
trapper:
   Exit Sub
End Sub
Private Sub Drive1_Change()
   On Error GoTo trapper
   ' Update directory list and file list when drive changes
   Dir1.Path = Drive1.Drive
   File1.Path = Dir1.Path
trapper:
```

```
    Exit Sub
End Sub
Private Sub SearchMacro(SearchString As String)
        On Error GoTo trapper
        'Search string scans returning byte locations of the matched string
        'This only finds the first occurance of the searchstring
        Bytematch = InStr(Filebuffer, SearchString)
        'If bytematch is not zero, a search string has been found
        If Bytematch > 0 Then
            'Loop this to find all searchstrings within the document and save locations
            Do While Bytematch > 0
                'Changing the matched byte in the memory copy of file allows the
                'InStr function to find another occurance of the same macroname
                Mid(Filebuffer, Bytematch, 1) = "x"
                'Give description of macro and location in program log box
                List2.AddItem SearchString & " macro found at location " & Bytematch
                'Update screen display now
                List2.Refresh
                'Add the filename and byte location to list of infected documemnts
                List3.AddItem Filename
                List3.AddItem Bytematch
                'Sets the flag to warn user at end of scan - a virus was found
                Virusfound = 1
                'Set flag to look for more macronames because it's possibly infected
                Virusfoundinthisfile = 1
                'Perform another search now that first occurance has been stored
                Bytematch = InStr(Filebuffer, SearchString)
            Loop
        End If
trapper:
    Exit Sub
End Sub
Private Sub File1_DBLClick()
    On Error GoTo trapper
    ' When a single file is double clicked in the filebox, it's added to selected files
    For i = 0 To File1.ListCount - 1
        If File1.Selected(i) = True Then
            If Right$(Dir1, 1) <> "\" Then
                List1.AddItem Dir1 + "\" + File1
            Else
                List1.AddItem Dir1 + File1
            End If
            Scan.Enabled = True
        End If
    Next i
trapper:
    Exit Sub
End Sub
Private Sub FileExit_Click()
    End                     ' Program terminates from the file,exit menu option
End Sub
Private Sub HelpIndex_Click()
    'Set the name of the help file
    CommonDialog1.HelpFile = "GW-SCAN.HLP"
    CommonDialog1.HelpCommand = cdlHelpContents
    CommonDialog1.HelpContext = 0
    ' Display Visual Basic Help contents topic.
    CommonDialog1.ShowHelp
End Sub
Private Sub HelpUserguide_Click()
```

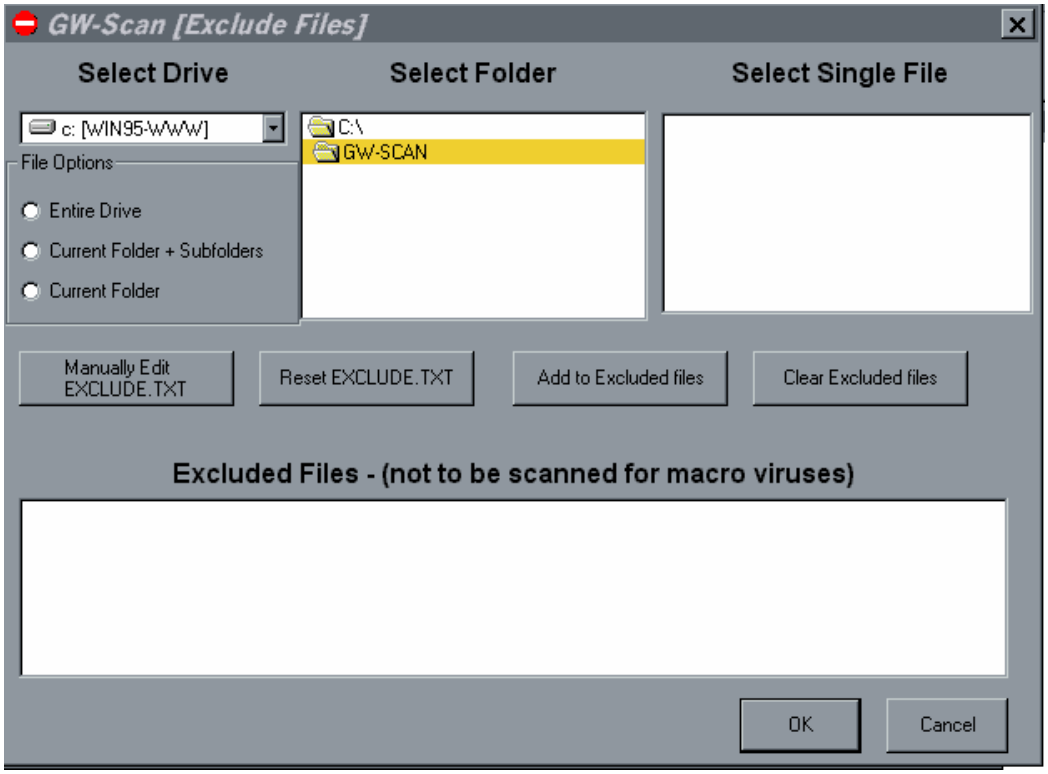
```
    'Set the name of the help file
    CommonDialog1.HelpFile = "USRGUIDE.HLP"
    CommonDialog1.HelpCommand = cdlHelpContents
    CommonDialog1.HelpContext = 0
    ' Display Visual Basic Help contents topic.
    CommonDialog1.ShowHelp
End Sub
Private Sub FileExclude_Click()
    Exclude.Show
    Filebox.Hide
    Exclude.List1.Clear
End Sub
Private Sub OptionsNormal_Click()
    Normal.Show
    Filebox.Hide
    Normal.Command1.Enabled = False
    Normal.List1.Clear
    Normal.List2.Clear
End Sub
Private Sub OptionsSearch_Click()
    Filebox.Hide
    Search.Show
End Sub
Private Sub Scan_Click()
    Dim Bytematch As Integer
    Waitbox.Show
    Waitbox.Refresh
    On Error GoTo trapper
    'If there are items in the selected files list then perform this if statement
    If List1.ListCount > 0 Then
    'For every file in the list box, peform everything within this for statement
    'Exclude procedure strips the files from list before scanning
    For i = 0 To List1.ListCount - 1
            Include = 1
            Open "C:\GW-Scan\EXCLUDE.TXT" For Input As #1
            Do Until EOF(1)
            Open "C:\GW-Scan\EXCLUDE.TXT" For Input As #1
            Input #1, A$
            'If the file is in exclude.txt then prevent a scan
            If UCase(A$) = UCase(List1.List(i)) Then Include = 0
            'Show in the program log the file was excluded
            If Include = 0 Then List2.AddItem "EXCLUDED:" + List1.List(i)
     Loop
Close #1

'Only scan if the file is to be included
If Include = 1 Then
'Show the filename that is currently being scanned
Text2.Text = List1.List(i)
'Display stright away
Text2.Refresh
' Set the current filename from the list of files to scan
            Filename = List1.List(i)
              'Add the filename into the log
              List2.AddItem Filename
              ' Open the document filename being processed
              Open Filename For Binary Access Read As #1
              Filebuffer = ""
              'Read entire file from the listbox into the string filebuffer
              Filebuffer = Input(LOF(1), #1)
```

```
            'File has been read to the filebuffer, now close file
            Close #1

            'Convert all lowercase letters in the file image to uppercase
            'This reduces the amount of search strings needed
            Filebuffer = UCase(Filebuffer)
            'Call search procedure to look for uppercase macronames
            Open "C:\GW-Scan\SEARCH.TXT" For Input As #1
            Do Until EOF(1)
               Input #1, A$
               SearchMacro (UCase(A$))
            Loop
            Close #1
         End If
      Next i
   End If

   'Remove the please wait message
   Waitbox.Hide

   'Search macros existed within the documents, so warn user of possible virus
   If Virusfound = 1 Then
      List3.Visible = True
      List2.AddItem "Automacros were found, press clean to remove them"
      List2.AddItem "Or press Clear Selected files to perform another scan"
      'Red background and change of text to alert user
      Label1.BackColor = &HFF&
      Label1 = "Infected Files"
      'Prevent re-scan of same files and allow user to clean them
      Scan.Enabled = False
      Clean.Enabled = True
   End If

   'Write the program log to REPORT.TXT C:\GW-Scan
   'This is overwritten each time the scanner runs
   Open "C:\GW-Scan\REPORT.TXT" For Output As #1

   For i = 0 To List2.ListCount - 1
      Print #1, List2.List(i)
   Next i

   Close #1
trapper:
   Exit Sub
End Sub
Private Sub Form_Load()

   'READ THE INI FILE
   ' Set the current filename from the list of files to scan
   Filename = "C:\WINDOWS\GW-SCAN.INI"

   ' Open the document filename being processed
   Open Filename For Binary Access Read As #1
   Filebuffer = ""

   'Read entire file from the initalization file into the string filebuffer
   Filebuffer = Input(LOF(1), #1)

   'File has been read to the filebuffer, now close file
   Close #1
```

```
    'Convert all lowercase letters in the file image to uppercase
    'This reduces the amount of search strings needed
    Filebuffer = UCase(Filebuffer)

    INIFile = Filebuffer
    Close #1
    'Show intro screen
    SplashScreenDisplay
    'Flag states no viruses have been found yet
    Virusfound = 0
End Sub
Private Sub HelpAbout_Click()
    SplashScreenDisplay     ' User clicked on about
                            ' show splashscreen
End Sub
Private Sub SplashScreenDisplay()
    Dim Start, Pausetime
    Pausetime = 5           ' Set duration of pause.
    Start = Timer           ' Set start time.
    SplashScreen.Show       ' Display SplashScreen Logo
    Do While Timer < Start + Pausetime
        DoEvents            ' Yield to other processes.
    Loop
    SplashScreen.Hide       ' Remove SplashScreen Logo
End Sub
```

Description : Filebox selector for GW-Scan, The Main Screen where all controls are located
Avaiability : After spalshscreen display, and when all subforms are closed & returns to Filebox
Form Design : The Invisible Man

The Exclude Form:



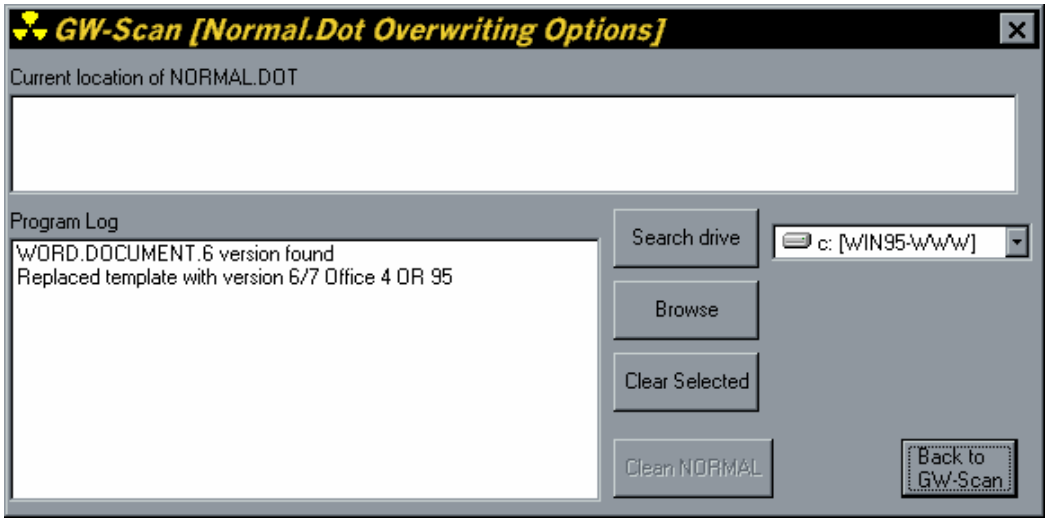

```
Open "C:\GW-Scan\EXCLUDE.TXT" For Input As #1
Private Sub ScanDirs(Toplevel As String, rListBox As ListBox)
'Recursively scan subdirs.
Dim strFiles() As String
Dim iCtr As Integer
If Right$(Toplevel, 1) <> "\" Then Toplevel = Toplevel & "\"
ReDim strFiles(0 To 1)
strFiles(0) = Dir$(Toplevel & "*.*", vbDirectory)
Do Until Len(strFiles(0)) = 0
   If strFiles(0) <> String$(Len(strFiles(0)), ".") Then
      strFiles(UBound(strFiles)) = IIf((GetAttr(Toplevel & strFiles(0)) And vbDirectory), "*", " ") &
strFiles(0)
      ReDim Preserve strFiles(UBound(strFiles) + 1)
   End If
   strFiles(0) = Dir$
Loop
For iCtr = 1 To UBound(strFiles) - 1
If (UCase$(Right$(strFiles(iCtr), 4)) = ".DOT" Or UCase$(Right$(strFiles(iCtr), 4)) = ".DOC") Then
rListBox.AddItem Toplevel & Right$(strFiles(iCtr), Len(strFiles(iCtr)) - 1)
If Left$(strFiles(iCtr), 1) = "*" Then
   Call ScanDirs(Toplevel & Right$(strFiles(iCtr), Len(strFiles(iCtr)) - 1), rListBox)
End If
Next iCtr
End Sub
Private Sub Command1_Click()
   'If the list is not empty, add to the exclude file
   On Error GoTo trapper
   If List1.ListCount > 0 Then
   'Add to the list of excluded files, or create file
      Open "C:\GW-Scan\EXCLUDE.TXT" For Append As #1
      For i = 0 To List1.ListCount - 1
```

```
            'Write path of file to exclude.txt
            Print #1, List1.List(i)
        Next i
        ' Close the exclude.txt file
        Close #1
    End If
trapper:
    Filebox.Show
    Unload Me
    Exit Sub
End Sub
Private Sub Command2_Click()
    Filebox.Show
    Unload Me
End Sub
Private Sub Command3_Click()
    Filebox.Show
    Unload Me
    Shell ("NOTEPAD.EXE C:\GW-Scan\EXCLUDE.TXT"), 1
End Sub
Private Sub Command4_Click()
    ' Set up the listbox variable
    Dim MyListBox As ListBox
    Set MyListBox = Exclude.List1
    ' Determine which file option is being used
    ' and decide if drives, directories or files are to be selected
    ' Select every document & template from the entire drive
    If Fileopt1.Value = True Then
        Waitbox.Show
        Waitbox.Refresh
        Call ScanDirs(Left$(Drive2.Drive, 2), MyListBox)
        Waitbox.Hide
    End If
    ' Select documents and templates from a directory and it's subdirectories
    If Fileopt2.Value = True Then
        Waitbox.Show
        Waitbox.Refresh
        Call ScanDirs(Dir2, MyListBox)
        Waitbox.Hide
    End If
    ' Select documents and templates from the current directory only
    If Fileopt3.Value = True Then
        Dim Pattern As String
        Dim FoundFile As String
        For i = 1 To 2
            If i = 1 Then Pattern = "*.DOC" Else Pattern = "*.DOT"
            ChDrive (Drive2)
            ChDir (Dir2)
            FoundFile = Dir$(Pattern)
            Do Until FoundFile = ""
                If Right$(Dir2, 1) <> "\" Then
                    MyListBox.AddItem Dir2 & "\" & FoundFile
                Else
                    MyListBox.AddItem Dir2 + FoundFile
                End If
                FoundFile = Dir$
            Loop
        Next i
    End If
End Sub
```

```
Private Sub Command5_Click()
     Exclude.List1.Clear         ' Clear contents of excluded files listbox
End Sub
Private Sub Command6_Click()
   'Resetting the exclude file will erase it's contents
   'And enable all documents to be scanned again
   On Error GoTo trapper
   Open "C:\GW-Scan\EXCLUDE.TXT" For Output As #1
   Close #1
   Exit Sub
trapper:
Exit Sub
End Sub
Private Sub Dir2_Change()
   On Error GoTo trapper
   ' Update file list when directory changes
   File2.Path = Dir2.Path
trapper:
   Exit Sub
End Sub
Private Sub Drive2_Change()
   'This error handler prevents a crash when accessing
   'drive A without a floppy disk in the drive
   On Error GoTo trapper
   ' Update directory list and file list when drive changes
   Dir2.Path = Drive2.Drive
   File2.Path = Dir2.Path
trapper:
   Exit Sub
End Sub
Private Sub File2_DBLClick()
   On Error GoTo trapper
   ' When a single file is double clicked in the filebox, it's added to selected files
   For i = 0 To File2.ListCount - 1
      If File2.Selected(i) = True Then
         If Right$(Dir1, 1) <> "\" Then
            List1.AddItem Dir2 + "\" + File2
         Else
            List1.AddItem Dir2 + File2
         End If
      End If
   Next i
trapper:
   Exit Sub
End Sub
```

Description : Same as filebox form with a manual edit facility
Availability : 1 User selects Exclude files from the Fie Menu
2 CTRL + E
Form Design : The Invisible Man

The Normal Form:




```
Dim Instpath As String                ' The directory containing GWscan and it's files
Dim Normalfilename, NFilebuffer  ' The normal templates filename and file in memory
Dim Bytematch As Integer              ' The location within a file of a matching string
Private Sub ScanDirs(Toplevel As String, rListBox As ListBox)
'Recursively scan subdir, but only looking for NORMAL.DOT
Dim strFiles() As String
Dim iCtr As Integer
On Error GoTo trapper
If Right$(Toplevel, 1) <> "\" Then Toplevel = Toplevel & "\"
ReDim strFiles(0 To 1)
strFiles(0) = Dir$(Toplevel & "*.*", vbDirectory)
Do Until Len(strFiles(0)) = 0
   If strFiles(0) <> String$(Len(strFiles(0)), ".") Then
      strFiles(UBound(strFiles)) = IIf((GetAttr(Toplevel & strFiles(0)) And vbDirectory), "*", " ") &
strFiles(0)
      ReDim Preserve strFiles(UBound(strFiles) + 1)
   End If
   strFiles(0) = Dir$
Loop
For iCtr = 1 To UBound(strFiles) - 1
If UCase$(Right$(strFiles(iCtr), 10)) = "NORMAL.DOT" Then rListBox.AddItem Toplevel &
Right$(strFiles(iCtr), Len(strFiles(iCtr)) - 1)
If Left$(strFiles(iCtr), 1) = "*" Then
   Call ScanDirs(Toplevel & Right$(strFiles(iCtr), Len(strFiles(iCtr)) - 1), rListBox)
End If
Next iCtr
trapper:
   Exit Sub
End Sub
Private Sub SearchVersion(SearchString As String)
'Search string scans returning byte locations of the matched string
'This only finds the first occurance of the searchstring
      Bytematch = InStr(NFilebuffer, SearchString)

      'If bytematch is not zero, a search string has been found
      If Bytematch > 0 Then
            'Find version number within the document and save locations
            'Give description of macro and location in program log box
```

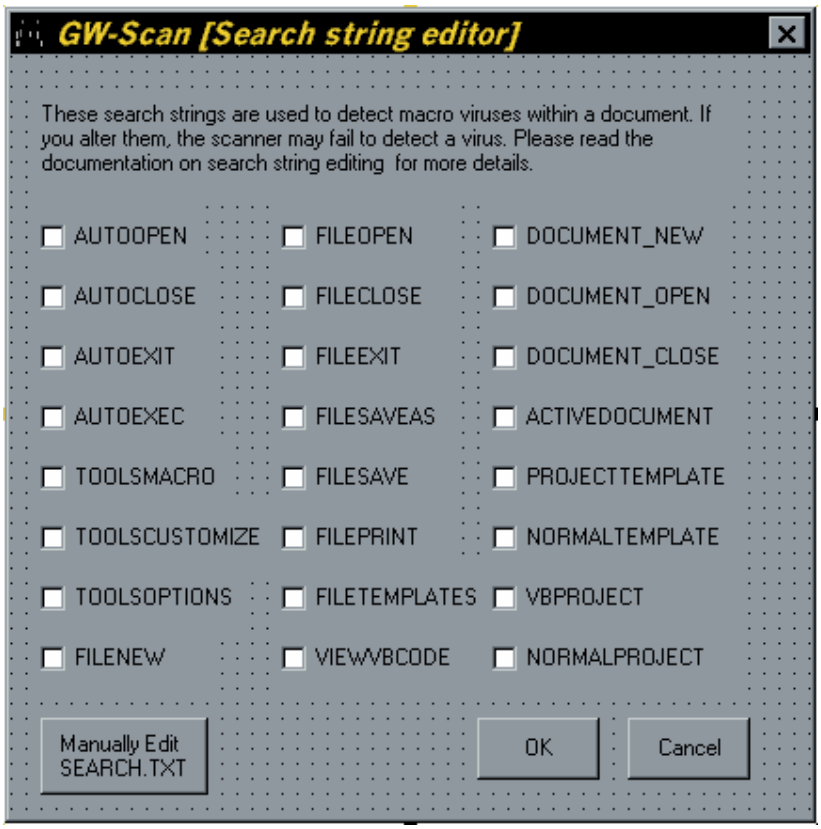
```
            List2.AddItem SearchString & " version found "
            'Update screen display now
            List2.Refresh
        End If
End Sub
Private Sub SearchNormal()
Dim OldName, NewName
    On Error GoTo trapper
    'If there are items in the selected files list then perform this if statement
    If List1.ListCount > 0 Then
    'For every file in the list box, peform for loop until all files are done
        For i = 0 To List1.ListCount - 1
            ' Set the current filename from the list of files to scan
            Normalfilename = List1.List(i)
            'Add the filename into the log
            'Open the document filename being processed
            Open Normalfilename For Binary Access Read As #1
            NFilebuffer = ""
            'Read entire file from the listbox into the string filebuffer
            Waitbox.Show
            Waitbox.Refresh
            NFilebuffer = Input(LOF(1), #1)
              'File has been read to the normal templates filebuffer
              'Convert all lowercase letters in the file image to uppercase
              'This reduces the amount of search strings needed
            NFilebuffer = UCase(NFilebuffer)

            Bytematch = 0
            'Call search procedure to look for uppercase version strings
            SearchVersion ("WORD.DOCUMENT.6") ' Word 6 / Office 4 & 95 stnd/pro

            'If version 6 or 7 not found, check for version 8
            If Bytematch = 0 Then
               SearchVersion ("WORD.DOCUMENT.8") ' Word 97 / Office 97 standard & pro
            End If
            Close #1

            'Remove the please wait message
            Waitbox.Hide

    'Next occurance of normal.dot to be version checked (in order of 6,7,8)
    'There could be all versions of normal.dot but these are all replaced
    'with the same version as the normal.dot which is overwritten
        Next i
    Command1.Enabled = False
End If
trapper:
    Exit Sub
End Sub
Private Sub Command1_Click()
    On Error GoTo trapper
    'Store names of the clean backup template files
    Dim w6name, w8name As String
    'Check if path has a forward slash and add one if absent
    If Right$(Instpath, 1) <> "\" Then Instpath = Instpath + "\"
    w6name = Instpath + "normal.6"
    w8name = Instpath + "normal.8"
    'for every copy of normal.dot,  Search for version 6,7,8
    'overwrite it with correct version of template & update program log
    For i = 0 To List1.ListCount - 1
```

```
        If InStr(List2.List(i), "DOCUMENT.6") <> 0 Then
            FileCopy w6name, List1.List(i)
            List2.AddItem "Replaced template with version 6/7 Office 4 OR 95"
        ElseIf InStr(List2.List(i), "DOCUMENT.8") <> 0 Then
            FileCopy w8name, List1.List(i)
            List2.AddItem "Replaced template with version 8 / Office 97"
        End If
    Next i
    'Finished copying over files, clear file list and remove cleaner button
    Command1.Enabled = False
    List1.Clear
trapper:
    Exit Sub
End Sub
Private Sub Command2_Click()
    Filebox.Show
    Unload Me
End Sub
Private Sub Command3_Click()
    Call Form_Load
End Sub
Private Sub Drive1_Change()
    On Error GoTo trapper
trapper:
    Exit Sub
End Sub
Private Sub Findnormal_Click() 'FINDS NORMAL.DOT ON CURRENT DRIVE AUTO
On Error GoTo trapper
    'Get list of filenames from selected disk drive
    List1.Clear
    List2.Clear
    Waitbox.Show
    Waitbox.Refresh
    'Begin search for main template file
    Call ScanDirs(Left$(Drive1.Drive, 2), List1)
    If List1.ListCount > 0 Then
        SearchNormal
        Command1.Enabled = True
    End If
    Waitbox.Hide
trapper:
    Exit Sub
End Sub
Private Sub Form_Load() 'NORMAL FORM JUST LOADED
    Command1.Enabled = False
    List1.Clear
    List2.Clear
    Instpath = "C:\GW-SCAN"
End Sub
Private Sub NormalButton_Click() 'BROWSING FOR NORMAL.DOT WITH COMMONDIALOG
    On Error GoTo trapper
    List1.Clear
    List2.Clear
    ' Set CancelError is True - quits back to main scanner
    With CommonDialog1
        .CancelError = True
        'Set flags to disable the 'open as read only' check box
        .Flags = cdlOFNHideReadOnly
        ' Set filters to show .dot extentions of normal.dot
        .Filter = "Normal Template (normal.dot)|normal.dot|Templates (*.dot)|*.dot"
```

```
        'Specify default filter
        .FilterIndex = 1
        ' Display the Open dialog box
        .ShowOpen
        ' Add name of selected file to a variable and list1
        Normalfilename = .Filename
    End With
    List1.AddItem Normalfilename
    If Normalfilename <> "" Then
        SearchNormal
        Command1.Enabled = True
    End If
    Exit Sub
trapper:
    'User pressed the Cancel button
    Exit Sub
End Sub
```

Description : Replacing Main Template Normal.dot with a known uninfected copy
Availavility : 1 When user selects Options menu, Replace Normal from the main screen
2 CTRL+R
Form Design : The Invisible Man

The Search String Form:

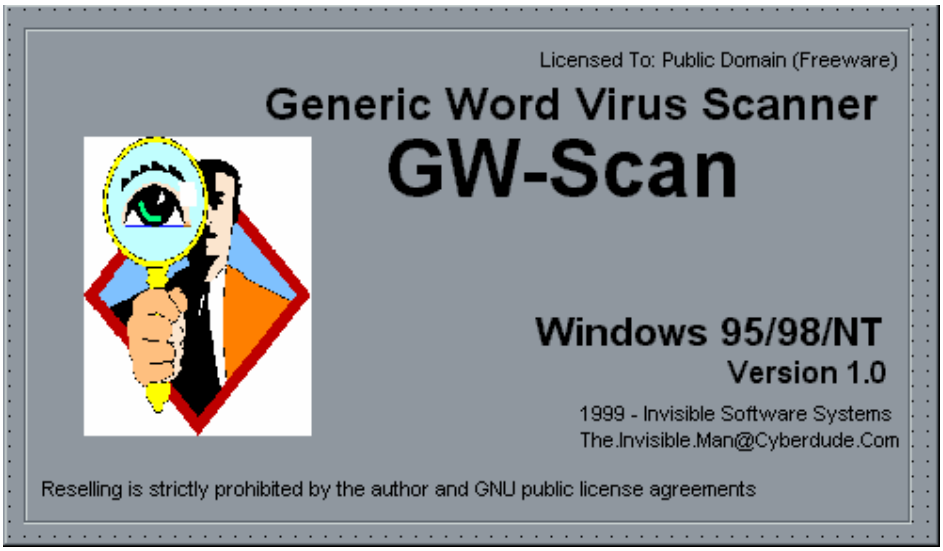



```
'This is the OK button option when a user has made their
'selection of search strings on the search form
'It will write the search strings to a search string file
Private Sub Command1_Click()
   'On any error, exit the whole module
   On Error GoTo trapper
   'Write to a new copy of the search string file
   Open "C:\GW-Scan\SEARCH.TXT" For Output As #1
   'For every option on the form, if it is checked
   'write the search string out to the search string file
   For i = 0 To 23
      If Check1(i).Value = 1 Then
         'The caption of the checkbox is the macroname
         Print #1, Check1(i).Caption
      End If
   Next i
   Close #1
   'Show main filebox form and unload the search module
   Filebox.Show
   Unload Me
trapper:
   Exit Sub
End Sub
Private Sub Command2_Click()
   'User pressed the cancel button, return to main screen
   Filebox.Show
   Unload Me
End Sub
```

```
Private Sub Command3_Click()
   'Manual edit of Search strings loads Notepad as editor
   Filebox.Show
   Unload Me
   On Error GoTo trapper
   Shell ("NOTEPAD.EXE C:\GW-Scan\SEARCH.TXT"), 1
trapper:
Exit Sub
End Sub
Private Sub Form_Load()
   'Reads in the search file when the form is first loaded
   'It then works out from the contents of the file which
   'Search checkboxes should be checked on the form
   On Error GoTo trapper
   For i = 0 To 23
   Open "C:\GW-Scan\SEARCH.TXT" For Input As #1
   Do Until EOF(1)
      Input #1, A$
      If A$ = Check1(i).Caption Then Check1(i) = 1
   Loop
   Close #1
   Next i
trapper:
   Exit Sub
End Sub
```

Description : Search string form, used to select or deselect virus srtings used for searches
Availability : 1 When user selects the Options menu, then Search string editor
2 CTRL+S
Form Design : The Invisible Man

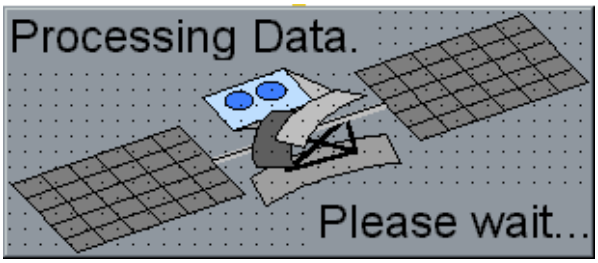
The Splashscreen Form: SplashScreen.Frm




```
Option Explicit

Private Sub Form_KeyPress(KeyAscii As Integer)
    Unload Me
End Sub
Private Sub Form_Load()
    lblVersion.Caption = "Version " & App.Major & "." & App.Minor & "." & App.Revision
    lblProductName.Caption = App.Title
End Sub
Private Sub Frame1_Click()
    Unload Me
End Sub
```

Description:	Spalshscreen loads to show program name, author and product information
Availability:	1 When user first loads program or when Help menu, About is used 2 CTRL + A
Picture:	Free from Mega Clipart 7000 CD-Rom – Royalty Free

Please wait box:Waitbox.Frm (no code)



Description: Displayed when the program is busy and the user has to be locked out from all commands
Availability: Produced when busy, disappears when given process is complete

Picture: Free from Mega Clipart 7000 CD-ROM – Royalty Free