



Visual Studio .NET

Source code exported to PDF

Solution name	DiskSize.sln
Main Project name	JadeDiskSize
Main Project location	Z:\Development\Dev\Products \VSNETcodePrint2005\Tests\DiskSize\DiskSize \DiskSize\JadeDiskSize.vbproj
Print date	07 October 2007
Author	Joginder S Nahil

Table of Contents

	Page	Line
DiskSize [Solution]		
JadeDiskSize [Project]	1	1
Filesystem.vb [ProjectItem]	1	1
Filesystem [Module]	1	5
DirectoryToXML [Function]	2	114
DirectoryToXML [Function]	2	121
GetDrives [Function]	2	85
GetUniversalName [Function]	1	35
REMOTE_NAME_INFO [Struct]	1	18
WriteDirectoryToXML [Function]	3	128
Form1.vb [ProjectItem]	5	1
frmDiskSize [Class]	5	4
AddTreeViewChildNodes [Function]	6	484
cboDrives_GotFocus [Function]	10	677
cboDrives_SelectedValueChanged [Function]	9	664
DrawPieChart [Function]	10	733
DriveToFilename [Function]	9	652
FindNode [Function]	10	681
Form1_Load [Function]	6	459
InitVars [Function]	5	431
LoadDriveList [Function]	6	464
LoadFromXML [Function]	9	657
menuItemBytes_Click [Function]	8	580
menuItemGBytes_Click [Function]	8	604
menuItemKBytes_Click [Function]	8	588
menuItemMBytes_Click [Function]	8	596
mnuExplore_Click [Function]	12	829
mnuOpen_Click [Function]	12	839
mnuRescanBranch_Click [Function]	12	825
picChart_MouseDown [Function]	11	739
picChart_MouseMove [Function]	13	903
RefreshTreeData [Function]	11	783
RefreshTreeDisplay [Function]	6	471
RefreshTreeDisplay [Function]	6	475
SetPiechartValues [Function]	10	709
Shell [Function]	12	850
SliceToNode [Function]	13	882
SortChildren [Function]	7	551
tbDrives_ButtonClick [Function]	13	873
tbFormToolbar_ButtonClick [Function]	8	612
tbGraph_ButtonClick [Function]	13	861
tvDirectory_AfterSelect [Function]	9	672

tvDirectory_MouseDown [Function]	11 -	768
XMLSort [Struct]	5 -	33
Icon1.ico [PhysicalFile]	15 -	1
My Project [PhysicalFolder]	15 -	1
NameValue.vb [ProjectItem]	15 -	1
NameValue [Class]	15 -	1
Name [Property]	15 -	17
New [Function]	15 -	5
ToString [Function]	15 -	23
Value [Property]	15 -	11
PieChart.vb [ProjectItem]	16 -	1
PieChart [Class]	16 -	3
AddValue [Function]	16 -	40
ClearValues [Function]	16 -	33
ConvertValuesToAngles [Function]	23 -	490
DarkShade [Function]	24 -	510
DrawChart [Function]	16 -	44
HighlightSlice [Function]	22 -	375
New [Function]	24 -	528
RedrawChart [Function]	23 -	453
SelectedSlice [Function]	20 -	280
SelectSlice [Function]	20 -	268
StringResources.vb [ProjectItem]	25 -	1
StringResources [Class]	25 -	6
GetString [Function]	25 -	25
New [Function]	25 -	10

```

00001 Imports System.IO
00002 Imports System.Runtime.InteropServices
00003 Imports System.Xml
00004
00005 Module Filesystem
00006     Public Enum Drivetypes
00007         Local = 1
00008         Network = 2
00009     End Enum
00010     Private Declare Auto Function WNetGetUniversalName Lib "mpr.dll" ( _
00011         <MarshalAs(System.Runtime.InteropServices.UnmanagedType.LPCTSTR)> _
00012         ByVal lpLocalPath As String, _
00013         ByVal dwInfoLevel As INFO_LEVEL, _
00014         ByVal lpBuffer As IntPtr, _
00015         ByRef lpBufferSize As Integer _
00016     ) As Integer
00017
00018     Private Structure REMOTE_NAME_INFO
00019         <MarshalAs(System.Runtime.InteropServices.UnmanagedType.LPCTSTR)> _
00020         Public lpUniversalName As String
00021         <MarshalAs(System.Runtime.InteropServices.UnmanagedType.LPCTSTR)> _
00022         Public lpConnectionName As String
00023         <MarshalAs(System.Runtime.InteropServices.UnmanagedType.LPCTSTR)> _
00024         Public lpRemainingPath As String
00025     End Structure
00026
00027     Private Enum INFO_LEVEL As Integer
00028         UNIVERSAL_NAME_INFO_LEVEL = 1
00029         REMOTE_NAME_INFO_LEVEL = 2
00030     End Enum
00031
00032     Private Const NO_ERROR = 0
00033     Private Const ERROR_MORE_DATA = 234
00034
00035     Public Function GetUniversalName( _
00036         ByVal Path As String, _
00037         ByRef UniversalName As String, _
00038         ByRef ConnectionName As String, _
00039         ByRef RemainingPath As String) As Boolean
00040
00041         ' When successful, returns TRUE with UniversalName,
00042         ' ConnectionName, and RemainingPath data. If not
00043         ' successful, it may be the drive is local and not mapped.
00044         Dim buffer As Integer
00045         Dim ptrbuffer As IntPtr
00046         Dim status As Integer
00047         Dim rni As REMOTE_NAME_INFO
00048         Dim Success As Boolean
00049         Dim SafetyCount As Integer = 0
00050
00051         UniversalName = ""
00052         ConnectionName = ""
00053         RemainingPath = ""
00054         buffer = 1024
00055
00056         ptrbuffer = Marshal.AllocHGlobal(buffer)
00057         status = WNetGetUniversalName(Path, INFO_LEVEL.REMOTE_NAME_INFO_LEVEL, ptrbuffer, buffer)
00058         Do While True
00059             Select Case status
00060                 Case NO_ERROR
00061                     rni = Marshal.PtrToStructure(ptrbuffer, GetType(REMOTE_NAME_INFO))
00062                     UniversalName = rni.lpUniversalName
00063                     ConnectionName = rni.lpConnectionName

```

1 2 3 4

```

00064 1 2 3 4   RemainingPath = rni.lpRemainingPath
00065         Success = True
00066         Exit Do
00067     } Case ERROR_MORE_DATA
00068     { If SafteyCount > 3 Then
00069         { Success = False
00070         { Exit Do
00071         { End If
00072         { SafteyCount += 1
00073         { Marshal.FreeHGlobal(ptrbuffer)
00074         { ptrbuffer = Marshal.AllocHGlobal(buffer)
00075         { status = WNetGetUniversalName(Path, INFO_LEVEL.REMOTE_NAME_INFO_LEVEL, ptrbuffer, buffer)
00076     } Case Else
00077         { Success = False
00078         { Exit Do
00079     } End Select
00080 } Loop
00081   Marshal.FreeHGlobal(ptrbuffer)
00082   Return Success
00083 End Function
00084
00085 Public Function GetDrives(ByVal DriveType As Drivetypes, ByVal LocalDiskText As String) As ArrayList
00086   Dim RtnVal As ArrayList = New ArrayList
00087   Dim DriveLetters As Array = ("A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z").Split(",")
00088   Dim DriveLetter As String
00089
00090   Dim UniversalName As String
00091   Dim ConnectionName As String
00092   Dim RemainingPath As String
00093
00094   For Each DriveLetter In DriveLetters
00095     DriveLetter = DriveLetter & "\\jade\$"
00096     If Directory.Exists(DriveLetter) Then
00097       If GetUniversalName(DriveLetter, UniversalName, ConnectionName, RemainingPath) Then
00098         If DriveType = Drivetypes.Network Then
00099           'This is a network drive and we are looking for network drives
00100           RtnVal.Add(New NameValue(DriveLetter, UniversalName & " (" & DriveLetter & ")"))
00101         End If
00102       Else
00103         If DriveType = Drivetypes.Local Then
00104           'This is a local drive and we are looking for local drives
00105           RtnVal.Add(New NameValue(DriveLetter, LocalDiskText & " (" & DriveLetter & ")"))
00106         End If
00107       End If
00108     End If
00109   Next
00110
00111   Return RtnVal
00112 End Function
00113
00114 Public Sub DirectoryToXML(ByVal sPath As String, ByRef XMLParentNode As XmlNode)
00115   Dim oXML As XmlNode = Nothing
00116   If Directory.Exists(sPath) Then
00117     WriteDirectoryToXML(sPath, XMLParentNode, Nothing)
00118   End If
00119 End Sub
00120
00121 Public Sub DirectoryToXML(ByVal sPath As String, ByRef XMLParentNode As XmlNode, ByRef
00122 updateBox As Label)
00122   Dim oXML As XmlNode = Nothing
00123   If Directory.Exists(sPath) Then
00124     WriteDirectoryToXML(sPath, XMLParentNode, updateBox)

```

```

00125 1 2 3 End If
00126 End Sub
00127
00128 Private Sub WriteDirectoryToXML(ByVal path As String, ByRef XMLParentNode As XmlNode, ByRef
    updateBox As Label)
00129     Dim oXML As XmlNode
00130     Dim oPathName As XmlAttribute
00131     Dim oFileSize As XmlAttribute
00132     Dim oSubDirSize As XmlAttribute
00133     Dim FileSize As Int64 = 0
00134     Dim SubDirSize As Int64 = 0
00135
00136     Dim source As DirectoryInfo = New DirectoryInfo(path)
00137     If (source.Exists) Then
00138         If Not (source.Attributes And System.IO.FileAttributes.Hidden) Then
00139             'create a node
00140             oXML = XMLParentNode.OwnerDocument.CreateElement("directory")
00141
00142             'Set the pathname attribute
00143             oPathName = XMLParentNode.OwnerDocument.CreateAttribute("pathname")
00144             oPathName.Value = path
00145             oXML.Attributes.Append(oPathName)
00146             XMLParentNode.AppendChild(oXML)
00147
00148             Try
00149                 Dim dir As DirectoryInfo
00150                 For Each dir In source.GetDirectories()
00151                     If Not updateBox Is Nothing Then
00152                         updateBox.Text = dir.FullName
00153                         updateBox.Refresh()
00154                     End If
00155                     WriteDirectoryToXML(dir.FullName, oXML, updateBox)
00156                 Next
00157             Catch ex As Exception
00158             End Try
00159
00160             Try
00161                 Dim file As FileInfo
00162                 For Each file In source.GetFiles()
00163                     FileSize = FileSize + file.Length
00164                 Next
00165             Catch ex As Exception
00166             End Try
00167
00168             'Get size of children
00169             Dim oChildNode As XmlNode
00170             For Each oChildNode In oXML.ChildNodes
00171                 SubDirSize = SubDirSize + _
00172                     Convert.ToInt64(oChildNode.Attributes("filesize").Value) + _
00173                     Convert.ToInt64(oChildNode.Attributes("subdirsize").Value)
00174             Next
00175
00176             'Add the filesize attribute
00177             oFileSize = XMLParentNode.OwnerDocument.CreateAttribute("filesize")
00178             oFileSize.Value = FileSize
00179             oXML.Attributes.Append(oFileSize)
00180
00181             'add the subdirectorysize attribute
00182             oSubDirSize = XMLParentNode.OwnerDocument.CreateAttribute("subdirsize")
00183             oSubDirSize.Value = SubDirSize
00184             oXML.Attributes.Append(oSubDirSize)
00185         End If
00186     End If
00187 End Sub
1

```

00188 |
00189 | End Module

```

00001 Imports System.Xml
00002 Imports System.IO
00003
00004 Public Class frmDiskSize
00005     Inherits System.Windows.Forms.Form
00006
00007     Private Enum RefreshTargets
00008         Drive = 1
00009         Directory = 2
00010     End Enum
00011
00012     Public Enum SortTypes
00013         AlphaAsc = 1
00014         AlphaDesc = 2
00015         NumericAsc = 3
00016         NumericDesc = 4
00017     End Enum
00018     Private Enum SizeDivisors
00019         Bytes = 1
00020         KBytes = 1024
00021         MBytes = 1024 * 1024
00022         GBytes = 1024 * 1024 * 1024
00023     End Enum
00024
00025     Private _RegreshTarget As RefreshTargets
00026     Private _SizeDivisor As SizeDivisors
00027     Private _SortType As SortTypes
00028     Private _TreeXML As XmlDocument
00029     Private _sr As StringResources
00030     Private _PieChart As PieChart
00031     Private _PieSelectedNode As TreeNode
00032
00033     Structure XMLSort
00034         Implements IComparable
00035         Public SortType As SortTypes
00036         Public Text As String
00037         Public TotalSize As Int64
00038         Public oXML As XmlNode
00039         Public Function CompareTo(ByVal obj As Object) As Integer Implements IComparable.CompareTo
00040             Dim XMLSort As XMLSort = obj
00041             Select Case SortType
00042                 Case SortTypes.AlphaAsc
00043                     Return Me.Text.CompareTo(XMLSort.Text)
00044                 Case SortTypes.AlphaDesc
00045                     Return XMLSort.Text.CompareTo(Me.Text)
00046                 Case SortTypes.NumericAsc
00047                     Return Me.TotalSize.CompareTo(XMLSort.TotalSize)
00048                 Case SortTypes.NumericDesc
00049                     Return XMLSort.TotalSize.CompareTo(Me.TotalSize)
00050             End Select
00051         End Function
00052     End Structure
00053
00054
00055
00431 Private Sub InitVars()
00432     _TreeXML = New XmlDocument
00433     _RegreshTarget = RefreshTargets.Drive
00434     _SizeDivisor = SizeDivisors.KBytes
00435     _SortType = SortTypes.AlphaAsc
00436     _sr = New StringResources
00437     _PieChart = New PieChart
00438
00439     Me.menuItemBytes.Text = _sr.GetString("Bytes")

```

1 2

```

00440 1 2 Me.menuItemKBytes.Text = _sr.GetString("KBytes")
00441 Me.menuItemMBytes.Text = _sr.GetString("MBytes")
00442 Me.menuItemGBytes.Text = _sr.GetString("GBytes")
00443
00444 Me.btnDisplaySizeDetail.Text = _sr.GetString("SizeDisplay") & " " & _sr.GetString("KBytes")
00445 Me.lblFolders.Text = _sr.GetString("FoldersTitle")
00446
00447 Me.btnDisplaySizeDetail.ToolTipText = _sr.GetString("ttSizeDisplay")
00448 Me.btnSortAlphaAsc.ToolTipText = _sr.GetString("ttbtnSortAlphaAsc")
00449 Me.btnSortAlphaDesc.ToolTipText = _sr.GetString("ttbtnSortAlphaDesc")
00450 Me.btnSortNumericAsc.ToolTipText = _sr.GetString("ttbtnSortNumericAsc")
00451 Me.btnSortNumericDesc.ToolTipText = _sr.GetString("ttbtnSortNumericDesc")
00452
00453 Me.btnRescan.ToolTipText = _sr.GetString("ttbtnRescan")
00454 Me.btnRefreshDrives.ToolTipText = _sr.GetString("ttbtnRefreshDrives")
00455 Me.lblGraph.Text = _sr.GetString("GraphTitle")
00456 Me.btnBack.ToolTipText = _sr.GetString("ttbtnBack")
00457 End Sub
00458
-----
00459 Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles MyBase.
Load
00460     LoadDriveList()
00461     picChart.BackColor = Me.BackColor
00462 End Sub
00463
-----
00464 Private Sub LoadDriveList()
00465     _RefreshTarget = RefreshTargets.Drive
00466     cboDrives.DataSource = GetDrives(Filesystem.Drivetypes.Local, _sr.GetString("LocalDisk"))
00467     cboDrives.DisplayMember = "Name"
00468     '   cboDrives.ValueMember = "Value"
00469 End Sub
00470
-----
00471 Private Sub RefreshTreeDisplay()
00472     RefreshTreeDisplay("")
00473 End Sub
00474
-----
00475 Private Sub RefreshTreeDisplay(ByVal sPath As String)
00476     If _TreeXML.ChildNodes.Count > 0 Then
00477         tvDirectory.Nodes.Clear()
00478         AddTreeViewChildNodes(tvDirectory.Nodes, _TreeXML.ChildNodes(0), sPath)
00479     End If
00480
00481     DrawPieChart(sPath)
00482 End Sub
00483
-----
00484 Private Sub AddTreeViewChildNodes(ByVal parent_nodes As TreeNodeCollection, ByVal xml_node As
XmlNode, ByVal ExpandedPath As String)
00485     Dim SizeAbbreviation As String
00486     Dim sFormatString As String = "0.00"
00487
00488     Select Case _SizeDivisor
00489     Case SizeDivisors.Bytes
00490         SizeAbbreviation = _sr.GetString("BytesAbbreviation")
00491         sFormatString = "0"
00492     Case SizeDivisors.KBytes
00493         SizeAbbreviation = _sr.GetString("KBytesAbbreviation")
00494     Case SizeDivisors.MBytes
00495         SizeAbbreviation = _sr.GetString("MBytesAbbreviation")
00496     Case SizeDivisors.GBytes
00497         SizeAbbreviation = _sr.GetString("GBytesAbbreviation")

```

```

00498 1 2 3 End Select
00499
00500 If xml_node.Name <> "root" Then
00501     Dim FileSize As Int64 = 0
00502     Dim FileDisplaySize As Double = 0
00503     Dim new_node As TreeNode
00504     Dim NodeName As String
00505
00506     FileSize = Convert.ToInt64(xml_node.Attributes("filesize").Value)
00507     FileDisplaySize = FileSize / _SizeDivisor
00508
00509     'Create a file node
00510     If FileDisplaySize > 0 Then
00511         NodeName = FileDisplaySize.ToString(sFormatString).PadLeft(15, " ")
00512         NodeName = NodeName & " " & SizeAbbreviation & " " & _sr.GetString("Files")
00513         new_node = parent_nodes.Add(NodeName)
00514         new_node.ImageIndex = 2 'Document
00515         new_node.SelectedImageIndex = 2 'Document
00516         new_node.Tag = xml_node.Attributes("pathname").Value
00517     End If
00518
00519     'Sort Children
00520     Dim sortedOutput As ArrayList
00521     Dim XMLNSort As XMLSort
00522     Dim FolderDisplaySize As Double
00523
00524     sortedOutput = SortChildren(xml_node)
00525     'Write the nodes to the treeview
00526
00527     For Each onv As Object In sortedOutput
00528         XMLNSort = CType(onv, XMLSort)
00529         Dim child_node As XmlNode = XMLNSort.oXML
00530         FolderDisplaySize = XMLNSort.TotalSize / _SizeDivisor
00531
00532         NodeName = FolderDisplaySize.ToString(sFormatString).PadLeft(15, " ") & _
00533             " " & SizeAbbreviation & " " & XMLNSort.Text
00534         new_node = parent_nodes.Add(NodeName)
00535         new_node.ImageIndex = 3 'Folder
00536         new_node.SelectedImageIndex = 4 'Open folder
00537         new_node.Tag = child_node.Attributes("pathname").Value
00538         If new_node.Tag = ExpandedPath Then
00539             'Expand the nodes up the tree
00540             new_node.EnsureVisible()
00541             new_node.Expand()
00542             tvDirectory.SelectedNode = new_node
00543         End If
00544         AddTreeViewChildNodes(new_node.Nodes, child_node, ExpandedPath)
00545     Next
00546 Else
00547     AddTreeViewChildNodes(parent_nodes, xml_node.ChildNodes(0), ExpandedPath)
00548 End If
00549 End Sub
00550
00551 Private Function SortChildren(ByVal oXMLNode As XmlNode) As ArrayList
00552
00553     Dim RtnVal As ArrayList = New ArrayList
00554     Dim PathSegments As Array
00555     Dim TotalSize As Int64
00556     Dim XMLNSort As XMLSort
00557     Dim FileSize As Int64 = 0
00558
00559     'Iterate through all nodes on this level and add them to array
00560     For Each child_node As XmlNode In oXMLNode.ChildNodes
00561         PathSegments = child_node.Attributes("pathname").Value.ToString.Split("\")

```

```

00562 1 2 3   FileSize = Convert.ToInt64(child_node.Attributes("filesize").Value)
00563       TotalSize = Convert.ToInt64(child_node.Attributes("subdirsizen").Value) + FileSize
00564
00565       XMLNSort.SortType = _SortType
00566       XMLNSort.TotalSize = TotalSize
00567       XMLNSort.Text = PathSegments(PathSegments.Length - 1)
00568       XMLNSort.oXML = child_node
00569
00570       RtnVal.Add(XMLNSort)
00571   Next child_node
00572
00573   'Sort the nodes
00574   RtnVal.Sort()
00575
00576   Return RtnVal
00577
00578 End Function
00579
-----
00580 Private Sub menuItemBytes_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
00581     menuItemBytes.Click
00582     Dim SelectedItem As MenuItem = CType(sender, MenuItem)
00583     Dim ItemText As String = SelectedItem.Text
00584     Me.btnDisplaySizeDetail.Text = _sr.GetString("SizeDisplay") & " " & _sr.GetString("Bytes")
00585     _SizeDivisor = SizeDivisors.Bytes
00586     RefreshTreeDisplay()
00587 End Sub
-----
00588 Private Sub menuItemKBytes_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
00589     menuItemKBytes.Click
00590     Dim SelectedItem As MenuItem = CType(sender, MenuItem)
00591     Dim ItemText As String = SelectedItem.Text
00592     Me.btnDisplaySizeDetail.Text = _sr.GetString("SizeDisplay") & " " & _sr.GetString("KBytes")
00593     _SizeDivisor = SizeDivisors.KBytes
00594     RefreshTreeDisplay()
00595 End Sub
-----
00596 Private Sub menuItemMBytes_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
00597     menuItemMBytes.Click
00598     Dim SelectedItem As MenuItem = CType(sender, MenuItem)
00599     Dim ItemText As String = SelectedItem.Text
00600     Me.btnDisplaySizeDetail.Text = _sr.GetString("SizeDisplay") & " " & _sr.GetString("MBytes")
00601     _SizeDivisor = SizeDivisors.MBytes
00602     RefreshTreeDisplay()
00603 End Sub
-----
00604 Private Sub menuItemGBytes_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
00605     menuItemGBytes.Click
00606     Dim SelectedItem As MenuItem = CType(sender, MenuItem)
00607     Dim ItemText As String = SelectedItem.Text
00608     Me.btnDisplaySizeDetail.Text = _sr.GetString("SizeDisplay") & " " & _sr.GetString("GBytes")
00609     _SizeDivisor = SizeDivisors.GBytes
00610     RefreshTreeDisplay()
00611 End Sub
-----
00612 Private Sub tbFormToolBar_ButtonClick(ByVal sender As System.Object, ByVal e As System.Windows.
00613     Forms.ToolBarButtonClickEventArgs) Handles tbFormToolBar.ButtonClick
00614     If e.Button Is btnSortNumericAsc Then
00615         If Not btnSortNumericAsc.Pushed Then
00616             btnSortAlphaAsc.Pushed = False
00617             btnSortAlphaDesc.Pushed = False

```

```

00617 1 2 3 4 btnSortNumericAsc.Pushed = True
00618      btnSortNumericDesc.Pushed = False
00619      _SortType = SortTypes.NumericAsc
00620      RefreshTreeDisplay()
00621      End If
00622  } ElseIf e.Button Is btnSortNumericDesc Then
00623      If Not btnSortNumericDesc.Pushed Then
00624          btnSortAlphaAsc.Pushed = False
00625          btnSortAlphaDesc.Pushed = False
00626          btnSortNumericAsc.Pushed = False
00627          btnSortNumericDesc.Pushed = True
00628          _SortType = SortTypes.NumericDesc
00629          RefreshTreeDisplay()
00630      End If
00631  } ElseIf e.Button Is btnSortAlphaAsc Then
00632      If Not btnSortAlphaAsc.Pushed Then
00633          btnSortAlphaAsc.Pushed = True
00634          btnSortAlphaDesc.Pushed = False
00635          btnSortNumericAsc.Pushed = False
00636          btnSortNumericDesc.Pushed = False
00637          _SortType = SortTypes.AlphaAsc
00638          RefreshTreeDisplay()
00639      End If
00640  } ElseIf e.Button Is btnSortAlphaDesc Then
00641      If Not btnSortAlphaDesc.Pushed Then
00642          btnSortAlphaAsc.Pushed = False
00643          btnSortAlphaDesc.Pushed = True
00644          btnSortNumericAsc.Pushed = False
00645          btnSortNumericDesc.Pushed = False
00646          _SortType = SortTypes.AlphaDesc
00647          RefreshTreeDisplay()
00648      End If
00649  End If
00650 End Sub
00651


---


00652 Private Function DriveToFilename(ByVal sPath As String)
00653     Dim PathSegments As Array = sPath.Replace(Path.VolumeSeparatorChar, "").Split(Path.DirectorySeparatorChar)
00654     Return PathSegments(0) & ".xml"
00655 End Function
00656


---


00657 Private Sub LoadFromXML(ByVal sPath As String)
00658     If File.Exists(sPath) Then
00659         _TreeXML.Load(sPath)
00660         RefreshTreeDisplay()
00661     End If
00662 End Sub
00663


---


00664 Private Sub cboDrives_SelectedValueChanged(ByVal sender As Object, ByVal e As System.EventArgs)
00665     Handles cboDrives.SelectedValueChanged
00666     Try
00667         LoadFromXML(DriveToFilename(cboDrives.SelectedValue))
00668     Catch ex As Exception
00669         'will fire this event when adding items to combobox
00670     End Try
00671 End Sub
00672


---


00672 Private Sub tvDirectory_AfterSelect(ByVal sender As Object, ByVal e As System.Windows.Forms.
00673     TreeViewEventArgs) Handles tvDirectory.AfterSelect
00674     _RegreshTarget = RefreshTargets.Directory
00675     DrawPieChart(tvDirectory.SelectedNode.Tag)
00676 End Sub

```

```

00676 1
00677  } Private Sub cboDrives_GotFocus(ByVal sender As Object, ByVal e As System.EventArgs) Handles
00678  »   cboDrives.GotFocus
00678  _   _RegreshTarget = RefreshTargets.Drive
00679  End Sub
00680
00681  Private Function FindNode(ByVal sPath As String) As XmlNode
00682  Dim PathSegments As Array = sPath.Split(Path.DirectorySeparatorChar)
00683  Dim PathLevel As Int16
00684  Dim oParentNode As XmlNode
00685  Dim PathString As String = ""
00686
00687  oParentNode = _TreeXML.ChildNodes(0)
00688  For PathLevel = 0 To PathSegments.Length - 1
00689  If PathString = "" Then
00690  PathString = PathSegments(PathLevel) & Path.DirectorySeparatorChar
00691  Else
00692  If PathLevel > 1 Then
00693  PathString = PathString & Path.DirectorySeparatorChar & PathSegments(PathLevel)
00694  Else
00695  PathString = PathString & PathSegments(PathLevel)
00696  End If
00697  End If
00698
00699  For Each oNode As XmlNode In oParentNode.ChildNodes
00700  If oNode.Attributes("pathname").Value = PathString Then
00701  oParentNode = oNode
00702  Exit For
00703  End If
00704  Next
00705  Next
00706  Return oParentNode
00707 End Function
00708
00709  Private Sub SetPiechartValues(ByVal sPath As String)
00710  Dim oXMLNode As XmlNode = FindNode(sPath)
00711  Dim sortedOutput As ArrayList = SortChildren(oXMLNode)
00712  Dim XMLNSort As XMLSort
00713  Dim FileSize As Int64 = 0
00714
00715  If sPath = "" Then
00716  oXMLNode = oXMLNode.ChildNodes(0)
00717  sortedOutput = SortChildren(oXMLNode)
00718  End If
00719
00720  FileSize = Convert.ToInt64(oXMLNode.Attributes("filesize").Value.ToString)
00721
00722  If FileSize > 0 Then
00723  _PieChart.AddValue(_sr.GetString("Files"), FileSize)
00724  End If
00725
00726  For Each onv As Object In sortedOutput
00727  XMLNSort = CType(onv, XMLSort)
00728  Dim child_node As XmlNode = XMLNSort.oXML
00729  _PieChart.AddValue(XMLNSort.Text, XMLNSort.TotalSize)
00730  Next
00731 End Sub
00732
00733  Private Sub DrawPieChart(ByVal sPath As String)
00734  _PieChart.ClearValues()
00735  SetPiechartValues(sPath)
1 2

```

```

00736 1 2  _PieChart.DrawChart(picChart)
00737      End Sub
00738
00739  Private Sub picChart_MouseDown(ByVal sender As System.Object, ByVal e As System.Windows.
    >      Forms.MouseEventArgs) Handles picChart.MouseDown
00740      ' See if this is the right button.
00741      Dim SelectedSlice As Int32 = _PieChart.SelectSlice(e.X, e.Y, picChart)
00742
00743      If e.Button = MouseButtons.Left Then
00744          Dim oCurrNode As TreeNode
00745          If SelectedSlice = -1 Then
00746              'None Selected
00747          Else
00748              oCurrNode = SliceToNode(SelectedSlice)
00749              tvDirectory.SelectedNode = oCurrNode
00750              oCurrNode.Expand()
00751              oCurrNode.EnsureVisible()
00752          End If
00753      ElseIf e.Button = MouseButtons.Right Then
00754          If SelectedSlice = -1 Then
00755              'None Selected
00756          Else
00757              If SelectedSlice = -1 Then
00758                  'None Selected
00759                  _PieSelectedNode = Nothing
00760              Else
00761                  _PieSelectedNode = SliceToNode(SelectedSlice)
00762                  cmTreeViewPopup.Show(picChart, New Point(e.X, e.Y))
00763              End If
00764          End If
00765      End If
00766      End Sub
00767
00768  Private Sub tvDirectory_MouseDown(ByVal sender As Object, ByVal e As System.Windows.Forms.
    >      MouseEventArgs) Handles tvDirectory.MouseDown
00769      ' See if this is the right button.
00770      If e.Button = MouseButtons.Right Then
00771          ' Select this node.
00772          Dim node_here As TreeNode = tvDirectory.GetNodeAt(e.X, e.Y)
00773          tvDirectory.SelectedNode = node_here
00774
00775          ' See if we got a node.
00776          If Not node_here Is Nothing Then
00777              _PieSelectedNode = Nothing
00778              cmTreeViewPopup.Show(tvDirectory, New Point(e.X, e.Y))
00779          End If
00780      End If
00781      End Sub
00782
00783  Private Sub RefreshTreeData()
00784      Dim oXML As XmlDocument
00785      Dim oXMLNode As XmlNode
00786
00787      Dim DriveLetter As String
00788      DriveLetter = cboDrives.SelectedValue
00789
00790      If tvDirectory.SelectedNode Is Nothing Then
00791          _RegreshTarget = RefreshTargets.Drive
00792      End If
00793
00794      If _RegreshTarget = RefreshTargets.Directory Then
00795          'Get the current selected node from the treeview

```

```

00796 1 2 3 Dim tvPath As String
00797     If Not _PieSelectedNode Is Nothing Then
00798         tvPath = _PieSelectedNode.Tag
00799     Else
00800         tvPath = tvDirectory.SelectedNode.Tag
00801     End If
00802     Dim oXMLFoundNode As XmlNode
00803     oXMLFoundNode = FindNode(tvPath)
00804     oXMLNode = oXMLFoundNode.ParentNode
00805     oXMLNode.RemoveChild(oXMLFoundNode)
00806     DirectoryToXML(tvPath, oXMLNode, lblStatus)
00807     _TreeXML.Save(DriveToFilename(DriveLetter))
00808     RefreshTreeDisplay(tvPath)
00809     'Open the treeview to the correct node
00810 } Else
00811     'Get the currently selected drive letter
00812     oXML = New XmlDocument
00813     oXMLNode = oXML.CreateElement("root")
00814     DirectoryToXML(DriveLetter, oXMLNode, lblStatus)
00815     oXML.AppendChild(oXMLNode)
00816     _TreeXML = oXML
00817     _TreeXML.Save(DriveToFilename(DriveLetter))
00818     RefreshTreeDisplay()
00819 End If
00820
00821     lblStatus.Text = ""
00822 End Sub
00823
00824


---


00825 Private Sub mnuRescanBranch_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
00826     mnuRescanBranch.Click
00827     RefreshTreeData()
00828 End Sub


---


00829 Private Sub mnuExplore_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
00830     mnuExplore.Click
00831     Dim tvPath As String
00832     If Not _PieSelectedNode Is Nothing Then
00833         tvPath = _PieSelectedNode.Tag
00834     Else
00835         tvPath = tvDirectory.SelectedNode.Tag
00836     End If
00837     Shell(tvPath, "explore")
00838 End Sub


---


00839 Private Sub mnuOpen_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles mnuOpen
00840     .Click
00841     Dim tvPath As String
00842     If Not _PieSelectedNode Is Nothing Then
00843         tvPath = _PieSelectedNode.Tag
00844     Else
00845         tvPath = tvDirectory.SelectedNode.Tag
00846     End If
00847     Shell(tvPath, "open")
00848 End Sub


---


00850 Private Sub Shell(ByVal sFilePath As String, ByVal sArgs As String)
00851     Dim m_oProc As Process = New Process
00852     Dim oInfo As ProcessStartInfo
00853     oInfo = New ProcessStartInfo(sFilePath)

```

```

00854 1 2 oInfo.UseShellExecute = True
00855 oInfo.Verb = sArgs
00856 oInfo.WindowStyle = ProcessWindowStyle.Normal
00857 m_oProc.StartInfo = oInfo
00858 m_oProc.Start()
00859 End Sub
00860

00861 Private Sub tbGraph_ButtonClick(ByVal sender As System.Object, ByVal e As System.Windows.Forms
> .ToolBarButtonClickEventArgs) Handles tbGraph.ButtonClick
00862     If Not tvDirectory.SelectedNode Is Nothing AndAlso Not tvDirectory.SelectedNode.Parent Is Nothing Then
00863         tvDirectory.SelectedNode = tvDirectory.SelectedNode.Parent
00864     ElseIf Not tvDirectory.SelectedNode Is Nothing AndAlso Not tvDirectory.SelectedNode.PrevNode Is Nothing Then
00865         Dim tvNode As TreeNode = tvDirectory.SelectedNode
00866         Do While Not tvNode.PrevNode Is Nothing
00867             tvNode = tvNode.PrevNode
00868         Loop
00869         tvDirectory.SelectedNode = tvNode
00870     End If
00871 End Sub
00872

00873 Private Sub tbDrives_ButtonClick(ByVal sender As System.Object, ByVal e As System.Windows.Forms
> .ToolBarButtonClickEventArgs) Handles tbDrives.ButtonClick
00874     If e.Button Is btnRefreshDrives Then
00875         LoadDriveList()
00876     ElseIf e.Button Is btnRescan Then
00877         _PieSelectedNode = Nothing
00878         RefreshTreeData()
00879     End If
00880 End Sub
00881

00882 Private Function SliceToNode(ByVal SliceNumber As Int32) As TreeNode
00883     Dim RtnVal As TreeNode
00884
00885     If tvDirectory.SelectedNode Is Nothing Then
00886         RtnVal = tvDirectory.Nodes(0)
00887     Else
00888         RtnVal = tvDirectory.SelectedNode
00889     End If
00890
00891     If RtnVal.Text.IndexOf(_sr.GetString("Files")) > -1 Then
00892         'We start at 1 here because we are already on the first node
00893         For NodeCounter As Int16 = 1 To SliceNumber
00894             RtnVal = RtnVal.NextNode
00895         Next
00896     Else
00897         RtnVal = RtnVal.Nodes(SliceNumber)
00898     End If
00899
00900     Return RtnVal
00901 End Function
00902

00903 Private Sub picChart_MouseMove(ByVal sender As Object, ByVal e As System.Windows.Forms.
> MouseEventArgs) Handles picChart.MouseMove
00904     Dim SelectedSlice As Int32 = _PieChart.SelectSlice(e.X, e.Y, picChart)
00905
00906     If SelectedSlice > -1 Then
00907         Dim oNode As TreeNode = SliceToNode(SelectedSlice)
00908         Static ToolTip As String
00909
00910         If ToolTip <> oNode.Text.Trim Then
00911             ToolTip = oNode.Text.Trim

```

```
00912 1 2 3 4 ttGraph.SetToolTip(picChart, ToolTip)
00913      End If
00914      End If
00915      End Sub
00916      End Class
```

```
00001 Public Class NameValue
00002     Private myValue As String
00003     Private myName As String
00004
00005     Public Sub New(ByVal Value As String, ByVal Name As String)
00006         MyBase.New()
00007         Me.myValue = Value
00008         Me.myName = Name
00009     End Sub
00010
00011     Public ReadOnly Property Value() As String
00012     {
00013         Get
00014             Return myValue
00015         End Get
00016     End Property
00017
00018     Public ReadOnly Property Name() As String
00019     {
00020         Get
00021             Return myName
00022         End Get
00023     End Property
00024
00025     Public Overrides Function ToString() As String
00026     {
00027         Return Me.Value & " - " & Me.Name
00028     End Function
00029 End Class
```

```

00001 Imports System.Drawing.Drawing2D
00002
00003 Public Class PieChart
00004     Private m_BitmapDark As Bitmap
00005     Private m_BitmapLight As Bitmap
00006     Private m_BitmapLabel As Bitmap
00007     Private m_HilightedBitmap As Bitmap = Nothing
00008
00009     Private m_names As ArrayList = New ArrayList
00010     Private m_values As ArrayList = New ArrayList
00011     Private m_lastSliceSelected As Int32 = -1
00012
00013     Private m_Angles() As Single = {0, 10, 45, 75, 100, 130, _
00014         170, 200, 225, 250, 270, 300, 315, 345, 360}
00015
00016     Private m_Colors() As Color = {Color.Blue, Color.Green, _
00017         Color.Cyan, Color.Red, Color.Magenta, Color.Yellow, _
00018         Color.BlueViolet, Color.Orange, Color.LightBlue, _
00019         Color.LightGreen, Color.LightCyan, Color.Pink, _
00020         Color.Maroon, Color.LightYellow, Color.SkyBlue}
00021
00022     Private m_sortAngles As ArrayList = New ArrayList
00023
00024     Private Declare Auto Function BitBlt Lib "gdi32.dll" (ByVal _
00025         hdcDest As IntPtr, ByVal nXDest As Integer, ByVal _
00026         nYDest As Integer, ByVal nWidth As Integer, ByVal _
00027         nHeight As Integer, ByVal hdcSrc As IntPtr, ByVal nXSrc _
00028         As Integer, ByVal nYSrc As Integer, ByVal dwRop As _
00029         System.Int32) As Boolean
00030     Private Const SRCCOPY As Integer = &HCC0020
00031
00032
00033     Public Sub ClearValues()
00034         m_names.Clear()
00035         m_values.Clear()
00036         m_lastSliceSelected = -1
00037         m_HilightedBitmap = Nothing
00038     End Sub
00039
00040     Public Sub AddValue(ByVal Name As String, ByVal Value As String)
00041         m_names.Add(Name)
00042         m_values.Add(Value)
00043     End Sub
00044
00045     Public Sub DrawChart(ByRef PictureBox As PictureBox)
00046         Dim ArrayAngles As Array = ConvertValuesToAngles(m_values.ToArray)
00047         'ArrayAngles = m_Angles
00048
00049         m_BitmapDark = Nothing
00050         m_BitmapLight = Nothing
00051         m_BitmapLabel = Nothing
00052
00053         Dim gr As Graphics = PictureBox.CreateGraphics
00054         ' See if we have a bitmap saved.
00055         ' Make the new bitmap.
00056         m_BitmapDark = New Bitmap( _
00057             PictureBox.ClientSize.Width, _
00058             PictureBox.ClientSize.Height, _
00059             gr)
00060         m_BitmapLight = New Bitmap( _
00061             PictureBox.ClientSize.Width, _

```

1 2

```

00062 1 2   PictureBox.ClientSize.Height, _
00063   gr)
00064   m_BitmapLabel = New Bitmap( _
00065   PictureBox.ClientSize.Width, _
00066   PictureBox.ClientSize.Height, _
00067   gr)
00068
00069   ' Draw on the bitmap.
00070   Dim pieSlice As Integer
00071   Dim bitmapDark_gr As Graphics = Graphics.FromImage(m_BitmapDark)
00072   Dim bitmapLight_gr As Graphics = Graphics.FromImage(m_BitmapLight)
00073   Dim bitmapLabel_gr As Graphics = Graphics.FromImage(m_BitmapLabel)
00074
00075   Dim AvailableHeight As Int32 = PictureBox.Height
00076   Dim AvailableWidth As Int32 = PictureBox.Width
00077
00078   'Since we want the pie to appear as if we are looking from an
00079   'upper side rather than directly overhead, we will make it an oval
00080   'by reducing the height more than the width
00081   'The pie will use 2/3 of the height
00082   Dim HeightRatio As Single = 3 / 5
00083   Dim PieHeightPortion As Int32 = AvailableHeight * HeightRatio
00084   'The pie will be 3/4 of the width
00085   Dim WidthRatio As Single = 5 / 8
00086   Dim PieWidthPortion As Int32 = AvailableWidth * WidthRatio
00087
00088   Dim VerticalCenterOffset As Int32 = (AvailableHeight - PieHeightPortion) / 2
00089   Dim HorizontalCenterOffset As Int32 = (AvailableWidth - PieWidthPortion) / 2
00090
00091   Dim PiePointX As Int32
00092   Dim PiePointY As Int32
00093
00094   'Specify how far from the center, each piece of the pie should be moved back
00095   Dim CenterSpaceRadius As Double = 10
00096
00097   'Since we are seperating the slices, make the slices
00098   'smaller so they will not move out of the picture
00099   '(if the pie is displayed in full height and width)
00100   Dim SliceOverhang As Int32 = CenterSpaceRadius * 2
00101
00102   'The depth of the sides of the chart
00103   Dim PieDepth As Int32 = 20
00104   Dim PieHeight As Int32 = PieHeightPortion - SliceOverhang - PieDepth
00105   Dim PieWidth As Int32 = PieWidthPortion - SliceOverhang
00106
00107   'Calculate the ratio between X and Y
00108   Dim xRatio As Single = 1
00109   Dim yRatio As Single = PieHeight / PieWidth
00110   If PieHeight > PieWidth Then
00111     xRatio = PieWidth / PieHeight
00112     yRatio = 1
00113   End If
00114
00115   Dim Angle As Int32
00116   Dim radians As Double
00117   Dim ColorValue As Int16
00118   Dim SliceCount As Int16 = 0
00119
00120   SliceCount = 0
00121   'Create the pies
00122   For pieSlice = ArrayAngles.GetLowerBound(0) To ArrayAngles.GetUpperBound(0) - 1
00123     If ArrayAngles(pieSlice + 1) <> ArrayAngles(pieSlice) Then
00124       'Find the angle for the center of the slice
00125       Angle = ArrayAngles(pieSlice) + ((ArrayAngles(pieSlice + 1) - ArrayAngles(pieSlice)) / 2)
00126       'Convert to radians

```

1 2 3 4

```

00127 1 2 3 4 radians = Angle / (180.0 / Math.PI)
00128
00129 'Find out what color we should use for this slice
00130 'When we run out of colors, we start from the beginning again
00131 ColorValue = SliceCount Mod (m_Colors.Length - 1)
00132
00133 'make a darker version of the current color
00134 'this darker color will be the sides
00135 Dim colr As Color = DarkShade(m_Colors(ColorValue))
00136
00137 'Calculate the XY point for the point of this slice
00138 PiePointX = Convert.ToInt32(CenterSpaceRadius * (Math.Cos(radians) * xRatio)) + CenterSpaceRadius +
    » HorizontalCenterOffset
00139 PiePointY = Convert.ToInt32(CenterSpaceRadius * (Math.Sin(radians) * yRatio)) + CenterSpaceRadius +
    » VerticalCenterOffset
00140
00141 'Draw all slices in the darker color
00142 bitmapDark_gr.FillPie( _
00143     New SolidBrush(colr), _
00144     PiePointX, _
00145     PiePointY, _
00146     PieWidth, _
00147     PieHeight, _
00148     ArrayAngles(pieSlice), ArrayAngles(pieSlice + 1) - ArrayAngles(pieSlice))
00149 SliceCount = SliceCount + 1
00150
00151 'Draw all slices in the lighter color
00152 'for the top
00153 bitmapLight_gr.FillPie( _
00154     New SolidBrush(m_Colors(ColorValue)), _
00155     PiePointX, _
00156     PiePointY, _
00157     PieWidth, _
00158     PieHeight, _
00159     ArrayAngles(pieSlice), ArrayAngles(pieSlice + 1) - ArrayAngles(pieSlice))
00160 End If
00161 Next
00162
00163 'Create the lines and labels
00164 Dim sliceDepth As Int32
00165 If PieHeight > PieWidth Then
00166     sliceDepth = PieHeight / 2
00167 Else
00168     sliceDepth = PieWidth / 2
00169 End If
00170
00171 Dim StartLineX As Int32
00172 Dim StartLineY As Int32
00173 Dim EndLineX1 As Int32
00174 Dim EndLineY1 As Int32
00175 Dim EndLineX2 As Int32
00176 Dim EndLineY2 As Int32
00177 Dim TextX As Int32
00178 Dim TextY As Int32
00179 Dim LastEndLineY2 As Int32 = 0
00180
00181 For pieSlice = ArrayAngles.GetLowerBound(0) To ArrayAngles.GetUpperBound(0) - 1
00182     'Distance between the edge of the pie and the start of the line
00183     'that radiates from the pie and connects the pie
00184     'with the label
00185     Dim StartPoint As Double = sliceDepth + 10
00186     'The length of the line
00187     Dim PointerLength As Double = 30
00188     Dim EndPoint As Double = StartPoint + PointerLength
00189

```

1 2 3

```

00190 1 2 3 'Only label if the slice is visible (not so small that it is less than 1 degree)
00191      If ArrayAngles(pieSlice + 1) <> ArrayAngles(pieSlice) Then
00192          'Find the angle for the center of the slice
00193          Angle = ArrayAngles(pieSlice) + ((ArrayAngles(pieSlice + 1) - ArrayAngles(pieSlice)) / 2)
00194
00195          'Convert to radians
00196          radians = Angle / (180.0 / Math.PI)
00197
00198          StartLineX = Convert.ToInt32(StartPoint * (Math.Cos(radians) * xRatio)) + CenterSpaceRadius +
          HorizontalCenterOffset
00199      »      StartLineY = Convert.ToInt32(StartPoint * (Math.Sin(radians) * yRatio)) + CenterSpaceRadius +
          VerticalCenterOffset
00200      »      EndLineX1 = Convert.ToInt32(EndPoint * (Math.Cos(radians) * xRatio)) + CenterSpaceRadius +
          HorizontalCenterOffset
00201      »      EndLineY1 = Convert.ToInt32(EndPoint * (Math.Sin(radians) * yRatio)) + CenterSpaceRadius +
          VerticalCenterOffset
00202
00203          StartLineX = StartLineX + (PieWidth / 2)
00204          StartLineY = StartLineY + (PieHeight / 2)
00205          EndLineX1 = EndLineX1 + (PieWidth / 2)
00206          EndLineY1 = EndLineY1 + (PieHeight / 2)
00207          EndLineX2 = EndLineX1
00208          EndLineY2 = EndLineY1
00209
00210          bitmapLabel_gr.DrawLine(Pens.DarkGray, StartLineX, StartLineY, EndLineX1, EndLineY1)
00211
00212          Dim larger As Int32
00213          Dim smaller As Int32
00214          Dim DirectionMultiplier As Int32
00215          If EndLineY1 > StartLineY Then
00216              DirectionMultiplier = 1
00217              larger = EndLineY1
00218              smaller = StartLineY
00219          Else
00220              DirectionMultiplier = -1
00221              larger = StartLineY
00222              smaller = EndLineY1
00223          End If
00224
00225          'How long we want the vertical part of the pointer line to be
00226          Dim EndLineLength As Int16 = 5
00227          'The distance between the end of the vertical line and the text
00228          Dim LineToTextDistance As Int16 = 3
00229          'provide the multiplier to calculate the distance between the
00230          'the start of the line and the end of the line (taking into account perspective)
00231          Dim perspectiveRatio As Single = (smaller / larger)
00232          'The height of the text
00233          Dim TextHeight As Int32 = 9
00234          'How far left we should offset the start of the text
00235          Dim TextSetback As Int32 = 20
00236          'The calculated total space from the endpoint
00237
00238          Dim lineoffset As Int32 = perspectiveRatio * EndLineLength * DirectionMultiplier
00239          Dim fontoffset As Int32
00240
00241          fontoffset = (TextHeight * 0.66) * DirectionMultiplier
00242          LineToTextDistance = LineToTextDistance * DirectionMultiplier
00243
00244          'Try to insure the 2 labels are not on top of each other
00245          Do Until Math.Abs(EndLineY2 - LastEndLineY2) > TextHeight - 2
00246              EndLineY2 = EndLineY2 + (1 * DirectionMultiplier)
00247          Loop
00248
00249          LastEndLineY2 = EndLineY2
00250 1 2 3 4

```

```

00251 1 2 3 4 EndLineY2 = EndLineY2 + lineoffset
00252 TextX = EndLineX2 - TextSetback
00253 TextY = EndLineY2 + lineoffset - fontoffset + LineToTextDistance
00254
00255 bitmapLabel_gr.DrawLine(Pens.DarkGray, EndLineX1, EndLineY1, EndLineX2, EndLineY2)
00256
00257 Dim fnt As Font = New Font(FontFamily.GenericSansSerif, TextHeight, FontStyle.Regular, Drawing.
    GraphicsUnit.Pixel)
00258 > bitmapLabel_gr.DrawString(m_names(pieSlice), fnt, Brushes.Black, Convert.ToSingle(TextX), Convert.
    ToSingle(TextY))
00259 > fnt.Dispose()
00260 End If
00261 Next
00262
00263 gr.Dispose()
00264
00265 RedrawChart(PictureBox)
00266 End Sub
00267

00268 Public Function SelectSlice(ByVal X As Int32, ByVal Y As Int32, ByRef PictureBox As PictureBox) As Int32
00269 Dim RtnVal As Int32 = SelectedSlice(X, Y, PictureBox)
00270
00271 If RtnVal <> m_lastSliceSelected Then
00272 m_HilghtedBitmap = Nothing
00273 RedrawChart(PictureBox)
00274 If RtnVal > -1 Then HighlightSlice(RtnVal, PictureBox)
00275 m_lastSliceSelected = RtnVal
00276 End If
00277 Return RtnVal
00278 End Function
00279

00280 Private Function SelectedSlice(ByVal X As Int32, ByVal Y As Int32, ByVal PictureBox As PictureBox) As
Int32
00281 > Dim I_Bitmap As Bitmap
00282 Dim ArrayAngles As Array = ConvertValuesToAngles(m_values.ToArray)
00283 'ArrayAngles = m_Angles
00284
00285 Dim gr As Graphics = PictureBox.CreateGraphics
00286 ' See if we have a bitmap saved.
00287
00288 ' Make the new bitmap.
00289 I_Bitmap = New Bitmap( _
00290 PictureBox.ClientSize.Width, _
00291 PictureBox.ClientSize.Height, _
00292 gr)
00293
00294 ' Draw on the bitmap.
00295 Dim pieSlice As Integer
00296 Dim bitmap_gr As Graphics = Graphics.FromImage(I_Bitmap)
00297
00298 Dim AvailableHeight As Int32 = PictureBox.Height
00299 Dim AvailableWidth As Int32 = PictureBox.Width
00300
00301 'Since we want the pie to appear as if we are looking from an
00302 'upper side rather than directly overhead, we will make it an oval
00303 'by reducing the height more than the width
00304 'The pie will use 2/3 of the height
00305 Dim HeightRatio As Single = 3 / 5
00306 Dim PieHeightPortion As Int32 = AvailableHeight * HeightRatio
00307 'The pie will be 3/4 of the width
00308 Dim WidthRatio As Single = 5 / 8
00309 Dim PieWidthPortion As Int32 = AvailableWidth * WidthRatio
00310
1 2

```

```

00311 1 2 Dim VerticalCenterOffset As Int32 = (AvailableHeight - PieHeightPortion) / 2
00312 Dim HorizontalCenterOffset As Int32 = (AvailableWidth - PieWidthPortion) / 2
00313
00314 Dim PiePointX As Int32
00315 Dim PiePointY As Int32
00316
00317 'Specify how far from the center, each piece of the pie should be moved back
00318 Dim CenterSpaceRadius As Double = 10
00319
00320 'Since we are seperating the slices, make the slices
00321 'smaller so they will not move out of the picture
00322 '(if the pie is displayed in full height and width)
00323 Dim SliceOverhang As Int32 = CenterSpaceRadius * 2
00324
00325 'The depth of the sides of the chart
00326 Dim PieDepth As Int32 = 20
00327 Dim PieHeight As Int32 = PieHeightPortion - SliceOverhang - PieDepth
00328 Dim PieWidth As Int32 = PieWidthPortion - SliceOverhang
00329
00330 'Calculate the ratio between X and Y
00331 Dim xRatio As Single = 1
00332 Dim yRatio As Single = PieHeight / PieWidth
00333 If PieHeight > PieWidth Then
00334     xRatio = PieWidth / PieHeight
00335     yRatio = 1
00336 End If
00337
00338 Dim Angle As Int32
00339 Dim radians As Double
00340 Dim RtnVal As Int32 = -1
00341
00342 For pieSlice = ArrayAngles.GetLowerBound(0) To ArrayAngles.GetUpperBound(0) - 1
00343     If ArrayAngles(pieSlice + 1) <> ArrayAngles(pieSlice) Then
00344         'Find the angle for the center of the slice
00345         Angle = ArrayAngles(pieSlice) + ((ArrayAngles(pieSlice + 1) - ArrayAngles(pieSlice)) / 2)
00346         'Convert to radians
00347         radians = Angle / (180.0 / Math.PI)
00348
00349         'Calclate the XY point for the point of this slice
00350         PiePointX = Convert.ToInt32(CenterSpaceRadius * (Math.Cos(radians) * xRatio)) + CenterSpaceRadius +
00351             HorizontalCenterOffset
00352         PiePointY = Convert.ToInt32(CenterSpaceRadius * (Math.Sin(radians) * yRatio)) + CenterSpaceRadius +
00353             VerticalCenterOffset
00354
00355         bitmap_gr.FillPie( _
00356             New SolidBrush(Color.Black), _
00357             PiePointX, _
00358             PiePointY, _
00359             PieWidth, _
00360             PieHeight, _
00361             ArrayAngles(pieSlice), ArrayAngles(pieSlice + 1) - ArrayAngles(pieSlice))
00362
00363         'See if the point at the X/Y coordinate is black
00364         If I_Bitmap.GetPixel(X, Y).ToArgb = Color.Black.ToArgb Then
00365             RtnVal = pieSlice
00366             Exit For
00367         End If
00368     End If
00369 Next
00370 gr.Dispose()
00371
00372 Return RtnVal
00373 End Function

```

```

00374 1
00375 Private Sub HighlightSlice(ByVal SelectedSlice As Int32, ByVal PictureBox As PictureBox)
00376     m_HilghitedBitmap = m_BitmapLight.Clone()
00377     Dim ArrayAngles As Array = ConvertValuesToAngles(m_values.ToArray)
00378     'ArrayAngles = m_Angles
00379
00380     Dim gr As Graphics = PictureBox.CreateGraphics
00381
00382     ' Draw on the bitmap.
00383     Dim pieSlice As Integer
00384     Dim bitmap_gr As Graphics = Graphics.FromImage(m_HilghitedBitmap)
00385
00386     Dim AvailableHeight As Int32 = PictureBox.Height
00387     Dim AvailableWidth As Int32 = PictureBox.Width
00388
00389     'Since we want the pie to appear as if we are looking from an
00390     'upper side rather than directly overhead, we will make it an oval
00391     'by reducing the height more than the width
00392     'The pie will use 2/3 of the height
00393     Dim HeightRatio As Single = 3 / 5
00394     Dim PieHeightPortion As Int32 = AvailableHeight * HeightRatio
00395     'The pie will be 3/4 of the width
00396     Dim WidthRatio As Single = 5 / 8
00397     Dim PieWidthPortion As Int32 = AvailableWidth * WidthRatio
00398
00399     Dim VerticalCenterOffset As Int32 = (AvailableHeight - PieHeightPortion) / 2
00400     Dim HorizontalCenterOffset As Int32 = (AvailableWidth - PieWidthPortion) / 2
00401
00402     Dim PiePointX As Int32
00403     Dim PiePointY As Int32
00404
00405     'Specify how far from the center, each piece of the pie should be moved back
00406     Dim CenterSpaceRadius As Double = 10
00407
00408     'Since we are seperating the slices, make the slices
00409     'smaller so they will not move out of the picture
00410     '(if the pie is displayed in full height and width)
00411     Dim SliceOverhang As Int32 = CenterSpaceRadius * 2
00412
00413     'The depth of the sides of the chart
00414     Dim PieDepth As Int32 = 20
00415     Dim PieHeight As Int32 = PieHeightPortion - SliceOverhang - PieDepth
00416     Dim PieWidth As Int32 = PieWidthPortion - SliceOverhang
00417
00418     'Calculate the ratio between X and Y
00419     Dim xRatio As Single = 1
00420     Dim yRatio As Single = PieHeight / PieWidth
00421     If PieHeight > PieWidth Then
00422         xRatio = PieWidth / PieHeight
00423         yRatio = 1
00424     End If
00425
00426     Dim Angle As Int32
00427     Dim radians As Double
00428     Dim RtnVal As Int32 = -1
00429
00430     pieSlice = SelectedSlice
00431     'Find the angle for the center of the slice
00432     Angle = ArrayAngles(pieSlice) + ((ArrayAngles(pieSlice + 1) - ArrayAngles(pieSlice)) / 2)
00433     'Convert to radians
00434     radians = Angle / (180.0 / Math.PI)
00435
00436     'Calculate the XY point for the point of this slice
00437     PiePointX = Convert.ToInt32(CenterSpaceRadius * (Math.Cos(radians) * xRatio)) + CenterSpaceRadius +
1 2

```

```

1 2
00438     HorizontalCenterOffset
    PiePointY = Convert.ToInt32(CenterSpaceRadius * (Math.Sin(radians) * yRatio)) + CenterSpaceRadius +
    VerticalCenterOffset
00439
00440     bitmap_gr.DrawPie( _
00441         Pens.Black, _
00442         PiePointX, _
00443         PiePointY, _
00444         PieWidth, _
00445         PieHeight, _
00446         ArrayAngles(pieSlice), ArrayAngles(pieSlice + 1) - ArrayAngles(pieSlice))
00447
00448     ' Draw the bitmap.
00449     gr.Dispose()
00450     RedrawChart(PicBox)
00451 End Sub
00452


---


00453 Public Sub RedrawChart(ByRef PicBox As PictureBox)
00454     Dim iPieThickness As Int16 = 10
00455     Dim gr As Graphics
00456
00457     'Clear the image
00458     gr = PicBox.CreateGraphics()
00459     Dim i_Bitmap As Bitmap = New Bitmap( _
00460         PicBox.ClientSize.Width, _
00461         PicBox.ClientSize.Height, _
00462         gr)
00463     PicBox.Image = i_Bitmap
00464     gr.Dispose()
00465
00466     Dim img As Image = PicBox.Image.Clone()
00467     gr = Graphics.FromImage(img)
00468
00469     For iPieLayer As Int16 = iPieThickness To 1 Step -1
00470         gr.DrawImage(m_BitmapDark, 0, 0 + iPieLayer)
00471
00472         If Not m_HilghitedBitmap Is Nothing Then
00473             gr.DrawImage(m_HilghitedBitmap, 0, 0 + iPieLayer - 1)
00474         End If
00475     Next
00476
00477     ' See if we have a highlighted bitmap saved.
00478     If Not m_HilghitedBitmap Is Nothing Then
00479         gr.DrawImage(m_HilghitedBitmap, 0, 0)
00480     ElseIf Not m_BitmapLight Is Nothing Then
00481         ' Draw the bitmap.
00482         gr.DrawImage(m_BitmapLight, 0, 0)
00483     End If
00484
00485     gr.DrawImage(m_BitmapLabel, 0, 0)
00486     gr.Dispose()
00487     PicBox.Image = img
00488 End Sub
00489


---


00490 Private Function ConvertValuesToAngles(ByVal ValueList As Array) As Array
00491     Dim TotalValue As Int64
00492     Dim RtnVal As ArrayList = New ArrayList
00493
00494     'Set the start degree
00495     RtnVal.Add(0)
00496
00497     For Each obj As Object In ValueList
00498         TotalValue = TotalValue + Convert.ToInt64(obj.ToString)

```

```

00499 1 2 3 Next
00500
00501     Dim Degree As Int16
00502     For Each obj As Object In ValueList
00503         Degree = Degree + (Convert.ToInt64(obj.ToString) / TotalValue) * 360
00504         RtnVal.Add(Degree)
00505     Next
00506
00507     Return RtnVal.ToArray
00508 End Function
00509


---


00510 Function DarkShade(ByVal Colr As Color) As Color
00511     Dim RtnVal As Color
00512     Dim DarkShadeAddValue As Int16 = -40
00513     Dim Red As Int32 = Colr.R + DarkShadeAddValue
00514     Dim Green As Int32 = Colr.G + DarkShadeAddValue
00515     Dim Blue As Int32 = Colr.B + DarkShadeAddValue
00516     If Red > 255 Then Red = 255
00517     If Green > 255 Then Green = 255
00518     If Blue > 255 Then Blue = 255
00519     If Red < 0 Then Red = 0
00520     If Green < 0 Then Green = 0
00521     If Blue < 0 Then Blue = 0
00522
00523     'Assign the color values to the color object
00524     RtnVal = Color.FromArgb(Colr.A, Red, Green, Blue)
00525     Return RtnVal
00526 End Function
00527


---


00528 Public Sub New()
00529     'Create a sorted list of angles.
00530     'Use this sort order when displaying the pie slices to insure a correct 3D look
00531
00532     For LeftSide As Int16 = 270 To 90 Step -1
00533         m_sortAngles.Add(LeftSide)
00534
00535         Dim RightSide As Int32 = (270 + (270 - LeftSide) + 1) Mod 361
00536         'Will only happen at 90 degrees (Leftside=90)
00537         If RightSide <> LeftSide Then
00538             m_sortAngles.Add(RightSide)
00539         End If
00540     Next
00541 End Sub
00542
00543 End Class

```

```
00001 Imports System
00002 Imports System.Globalization
00003 Imports System.Threading
00004 Imports System.Resources
00005 Imports System.Reflection

00006 Public Class StringResources
00007
00008     Private _ResourceSet As ResourceSet
00009
00010     Public Sub New()
00011         ' Get the culture of the currently executing thread.
00012         ' The value of ci will determine the culture of
00013         ' the resources that the resource manager retrieves.
00014         Dim ci As CultureInfo = Thread.CurrentThread.CurrentCulture
00015
00016         Dim rm As ResourceManager
00017         Try
00018             rm = New ResourceManager("DiskSize.strings." & ci.Name, [Assembly].GetExecutingAssembly.
00019                 GetSatelliteAssembly(ci))
00019         Catch ex As Exception
00020             ci = New CultureInfo("en-US")
00021             rm = New ResourceManager("DiskSize.strings." & ci.Name, [Assembly].GetExecutingAssembly.
00022                 GetSatelliteAssembly(ci))
00022         End Try
00023         _ResourceSet = rm.GetResourceSet(ci, True, True)
00024     End Sub

00025     Public Function GetString(ByVal Name As String) As String
00026         Return _ResourceSet.GetString(Name)
00027     End Function
00028 End Class
```

Index

A 1-3, 5-13, 15-25
 Abs 19
 Add 2, 6, 7, 10, 13, 16, 23
 AddTreeViewChildNodes 6, 7
 AddValue 10, 16, 24
 AfterSelect 9
 AllocHGlobal 1
 AlphaAsc 5, 8
 AlphaDesc 5, 8
 Angle 16-18, 20-23
 Append 3, 12
 AppendChild 3, 12
 Array 2, 7, 9, 10, 16, 17, 19-22, 24
 ArrayAngles 16, 17, 19-22
 ArrayList 2, 7, 10, 16, 23
 Attributes 3, 7, 10
 AvailableHeight 17, 20, 22
 AvailableWidth 17, 20, 22
 B 1-3, 5-13, 15-24
 BackColor 6
 BitBit 16
 Bitmap 16-21, 23
 bitmap_gr 20, 22
 bitmapDark_gr 17, 18
 bitmapLabel_gr 17, 19
 bitmapLight_gr 17, 18
 Black 20, 21, 23
 Blue 16, 24
 BlueViolet 16
 Brushes 20
 btnBack 6
 btnDisplaySizeDetail 6, 8
 btnRefreshDrives 6, 13
 btnRescan 6, 13
 btnSortAlphaAsc 6, 8
 btnSortAlphaDesc 6, 8
 btnSortNumericAsc 6, 8
 btnSortNumericDesc 6, 9
 buffer 1, 2
 Button 8, 11, 13
 ButtonClick 8, 13
 Bytes 5, 6, 8
 cboDrives 6, 9, 11
 cboDrives_GotFocus 10
 cboDrives_SelectedValueChanged 9
 CenterSpaceRadius 17, 19, 21, 22
 child_node 7, 10
 ChildNodes 3, 6, 7, 10
 ci 25
 Clear 6, 10, 16
 ClearValues 10, 16
 Click 8, 12
 ClientSize 16, 20, 23
 Clone 22, 23
 cmTreeViewPopup 11
 Color 6, 16, 17, 21, 24
 ColorValue 17
 colr 18, 24
 Colr 18, 24
 CompareTo 5
 ConnectionName 1, 2
 Convert 3, 7, 10, 16, 18-22, 24
 ConvertValuesToAngles 16, 20, 22, 23

Cos 18, 19, 21, 22
 Count 1, 6, 13, 17
 CreateAttribute 3
 CreateElement 3, 12
 CreateGraphics 16, 20, 22, 23
 CultureInfo 25
 CurrentCulture 25
 CurrentThread 25
 Cyan 16
 DarkGray 19
 DarkShade 18, 24
 DarkShadeAddValue 24
 DataSource 6
 Degree 24
 dir 2, 3, 5-7, 9, 11-13, 19
 DirectionMultiplier 19
 Directory 2, 5-7, 9, 11-13
 DirectoryInfo 3
 DirectorySeparatorChar 9
 DirectoryToXML 2, 12
 Filesystem 1, 3, 6
 frmDiskSize 5
 NameValue 2, 15
 PieChart 5, 6, 9, 10, 13, 16, 18, 20, 22, 24
 StringResources 5, 25
 DisplayMember 6
 Dispose 20, 21, 23
 DrawChart 11, 16, 20, 23
 DrawImage 23
 Drawing 16, 20
 Drawing2D 16
 DrawLine 19
 DrawPie 6, 9, 23
 DrawPieChart 6, 9
 DrawString 20
 Drive 1, 2, 5, 6, 9, 11, 13
 DriveLetter 2, 11
 DriveLetters 2
 DriveToFilename 9, 12
 DriveType 1, 2, 6
 Drivetypes 1, 2, 6
 dwInfoLevel 1
 dwRop 16
 e 1-4, 6-14, 16-25
 EndLineLength 19
 EndLineX1 18, 20
 EndLineX2 18, 20
 EndLineY1 18, 20
 EndLineY2 18, 19
 EndPoint 18
 EnsureVisible 7, 11
 ERROR_MORE_DATA 1
 EventArgs 6, 8, 9, 11-13
 ex 2, 3, 5-10, 12, 13, 18, 19, 21, 23, 24
 Exception 3, 9, 25
 Exists 2, 9
 Expand 6, 11
 ExpandedPath 6
 file 1, 3, 6, 7, 9, 10, 12
 File 1, 3, 6, 7, 9, 10, 12
 FileAttributes 3
 FileDisplaySize 7
 FileInfo 3
 FileSize 3, 7, 10
 Filesystem 1, 3, 6

FillPie	18, 21	LastEndLineY2	18, 19
FindNode	10, 12	lblFolders	6
fmt	20	lblGraph	6
FolderDisplaySize	7	lblStatus	12
Font	19	Left	7, 11, 24
FontFamily	20	LeftSide	24
fontoffset	19	Length	3, 8, 10, 18, 19, 26
FontStyle	20	LightBlue	16
Form	5-9, 11, 13	LightCyan	16
Form1_Load	6	LightGreen	16
Forms	5, 8, 11, 13	LightYellow	16
FreeHGlobal	2	lineoffset	19
frmDiskSize	5, 26	LineToTextDistance	19
FromArgb	24	Load	6, 9, 13
FromImage	17, 20, 22, 23	LoadDriveList	6, 13
FullName	3	LoadFromXML	9
G	1-3, 5-13, 15-24, 26	Local	1, 2, 6
GBytes	5, 6, 8	LocalDiskText	2
GenericSansSerif	20	lpBuffer	1
GetDirectories	3	lpBufferSize	1
GetDrives	2, 6	lpConnectionName	1
GetExecutingAssembly	25	lpLocalPath	1
GetFiles	3	lpRemainingPath	1
GetLowerBound	17, 21	LPTStr	1
GetNodeAt	11	lpUniversalName	1
GetPixel	21	m_Angles	16
GetResourceSet	25	m_BitmapDark	16, 17, 23
GetSatelliteAssembly	25	m_BitmapLabel	16, 17, 23
GetString	5, 7, 8, 10, 13, 25	m_BitmapLight	16, 17, 22, 23
GetUniversalName	1, 2	m_Colors	16, 18
GetUpperBound	17, 21	m_HilitedBitmap	16, 20, 22, 23
Globalization	25	m_lastSliceSelected	16, 20
GotFocus	10, 26	m_names	16, 20
gr	5, 6, 9, 11, 13, 16, 18-21, 23-26	m_oProc	12
Graphics	16, 20, 22, 23, 26	m_sortAngles	16, 24
GraphicsUnit	20	m_values	16, 20, 22
Green	16, 24	Magenta	16
hdcDest	16	Maroon	16
hdcSrc	16	Marshal	1, 2
Height	16, 17, 19-21, 23, 26	MarshalAs	1
HeightRatio	17, 20, 22	Math	18, 19, 21, 22
Hidden	3	MBytes	5, 6, 8
HighlightSlice	20, 22	MenuItem	5, 8
HorizontalCenterOffset	17, 19, 21, 22	menuItemBytes	5, 8
i_Bitmap	23	menuItemBytes_Click	8
IComparable	5	menuItemGBytes	6, 8
Image	7, 17, 20, 22, 23, 26	menuItemGBytes_Click	8
ImageIndex	7	menuItemKBytes	6, 8
img	23	menuItemKBytes_Click	8
IndexOf	13	menuItemMBytes	6, 8
INFO_LEVEL	1	menuItemMBytes_Click	8
InitVars	5	mnuExplore	12
Int16	10, 13, 17, 19, 23, 24	mnuExplore_Click	12
Int32	11, 13, 16-20, 22, 24	mnuOpen	12
Int64	3, 5, 7, 10, 23	mnuOpen_Click	12
InteropServices	1	mnuRescanBranch	12
IntPtr	1, 16	mnuRescanBranch_Click	12
IO	1-3, 5-7, 9, 13, 15, 17, 19-23, 25, 26	MouseButtons	11
iPieLayer	23	MouseDown	11
iPieThickness	23	MouseEventArgs	11, 13
ItemText	8	MouseMove	13
KBytes	5, 6, 8	myName	15
L_Bitmap	20, 21	myValue	15
Label	2, 16, 17, 19, 23, 26	Name	1-3, 7, 9, 12, 15, 16, 20, 25, 26
larger	19	NameValue	2, 15, 26

Network		1, 2	Process	12
New	2, 5, 7, 11, 12, 15, 16, 18, 20, 21, 23, 24		ProcessStartInfo	12
new_node		7	ProcessWindowState	13
NextNode		13	ptrbuffer	1
nHeight		16	PtrToStructure	1
NO_ERROR		1	Pushed	8
node_here		11	R	1-3, 5-13, 15-27
NodeCounter		13	radians	17, 19, 21, 22
NodeName		7	Red	16, 20, 23, 24
Nodes	3, 6, 7, 10, 13, 26		RedrawChart	20, 23
Normal		13	Reflection	25
NumericAsc		5, 8, 26	Refresh	3, 5, 6, 8, 9, 11, 13, 26
NumericDesc		5, 9, 26	RefreshTargets	5, 6, 9, 11
nWidth		16	RefreshTreeData	11, 13
nXDest		16	RefreshTreeDisplay	6, 8, 9, 12
nXSrc		16	Regular	20
nYDest		16	RemainingPath	1, 2
nYSrc		16	REMOTE_NAME_INFO	1
obj	5, 7-10, 12, 13, 23		REMOTE_NAME_INFO_LEVEL	1
Object	5, 7-10, 12, 13, 23		RemoveChild	12
oChildNode		3	Replace	9
oCurrNode		11	ResourceManager	25
oFileSize		3	Resources	5, 25-27
oInfo		12	ResourceSet	25
oNode		10, 11, 13	Right	11, 24
onv	3, 7, 10, 16, 18-22, 24, 26		RightSide	24
oParentNode		10	rm	5-9, 11, 13, 25, 26
oPathName		3	rni	1
Orange		16	RtnVal	2, 7, 13, 20-23
oSubDirSize		3	Runtime	1
OwnerDocument		3	SafetyCount	1
oXML	2, 3, 5, 7, 10, 11, 26		sArgs	12
oXMLFoundNode		12	Save	12
oXMLNode		7, 10, 11	SelectedImageIndex	7
PadLeft		7	SelectedItem	8
Parent	2, 3, 6, 10, 12, 13		SelectedNode	5, 7, 9, 11, 12
parent_nodes		6	SelectedSlice	11, 13, 20, 22
ParentNode		2, 3, 10, 12	SelectedValue	9, 11, 26
path	1, 2, 6, 7, 9, 10, 12, 26		SelectedValueChanged	9, 26
Path	1, 2, 6, 7, 9, 10, 12, 26		SelectSlice	11, 13, 20
PathLevel		10	sender	6, 8, 9, 11-13
PathSegments		7, 9	SetPiechartValues	10
PathString		10	SetToolTip	14
Pens		19, 23	sFilePath	12
perspectiveRatio		19	sFormatString	6
PI	5, 6, 9, 10, 12, 13, 16-23, 26, 27		Shell	12
PictureBox		16, 20, 22, 23	Show	11
picChart		6, 11, 13	Sin	16, 17, 19, 20, 22
picChart_MouseDown		11	SizeAbbreviation	6
picChart_MouseMove		13	SizeDivisors	5, 6, 8
PictureBox		16, 20, 22, 23	SkyBlue	16
PieChart	5, 6, 9, 10, 13, 16, 18, 20, 22, 24, 26		SliceCount	17
PieDepth		17, 21, 22	sliceDepth	18
PieHeight		17, 18, 20, 22	SliceNumber	13
PieHeightPortion		17, 20, 22	SliceOverhang	17, 21, 22
PiePointX		17, 18, 21, 22	SliceToNode	11, 13
PiePointY		17, 18, 21, 22	smaller	19
pieSlice		17, 18, 20-22	SolidBrush	18, 21
PieWidth		17, 18, 20, 22	Sort	5, 7, 8, 10, 16, 24, 26
PieWidthPortion		17, 20, 22	SortChildren	7, 10
Pink		16	sortedOutput	7, 10
Pixel		20, 21	SortType	5, 8, 9
Point	11, 17, 18, 21, 22, 26		SortTypes	5, 9
PointerLength		18	source	3, 5, 25-27
PrevNode		13	sPath	2, 6, 9, 10

JadeDiskSize

Split	2, 7, 9	xml_node	6
SRCCOPY	16	XmlAttribute	3
Start	12, 18	XmlDocument	5, 11
StartInfo	12	XmlNode	2, 5-7, 10, 11
StartLineX	18	XMLNSort	7, 10
StartLineY	18	XMLParentNode	2, 3
StartPoint	18	XMLSort	5, 7, 10
status	1, 12	xRatio	17, 19, 21, 22
StringResources	5, 25, 26	Y	1-3, 5-13, 15-24, 26-28
SubDirSize	3	Yellow	16
Success	1	yRatio	17, 19, 21, 22
System	1, 3, 5, 6, 8, 9, 11-13, 16, 25, 26		
Tag	7, 9, 12		
tbDrives	13		
tbDrives_ButtonClick	13		
tbFormToolbar	8		
tbFormToolbar_ButtonClick	8		
tbGraph	13		
tbGraph_ButtonClick	13		
Text	2, 3, 5, 7, 8, 10, 12, 13, 18, 19		
TextHeight	19		
TextSetback	19		
TextX	18, 20		
TextY	18, 20		
Thread	25, 26		
Threading	25		
ToArgb	21		
ToArray	16, 20, 22, 24		
ToInt32	18, 19, 21, 22		
ToInt64	3, 7, 10, 23		
ToolBarButtonClickEventArgs	8, 13		
ToolTip	6, 13		
ToolTipText	6		
ToSingle	20		
ToString	7, 10, 15, 23		
TotalSize	5, 7, 10		
TotalValue	23		
TreeNode	5, 6, 11, 13		
TreeNodeCollection	6		
TreeViewEventArgs	9		
Trim	13		
ttGraph	14		
tvDirectory	6, 7, 9, 11, 12		
tvDirectory_AfterSelect	9		
tvDirectory_MouseDown	11		
tvNode	13		
tvPath	12		
UNIVERSAL_NAME_INFO_LEVEL	1		
UniversalName	1, 2		
UnmanagedType	1		
updateBox	2		
UseShellExecute	13		
Value	2, 3, 7, 9-11, 15-17, 20, 22, 23, 26-28		
ValueList	23		
Verb	13		
VerticalCenterOffset	17, 19, 21, 22		
VolumeSeparatorChar	9		
Width	16-18, 20-22, 26		
WidthRatio	17, 20, 22		
Windows	5, 8, 9, 11, 13		
WindowState	13		
WNetGetUniversalName	1, 2		
WriteDirectoryToXML	2		
X	1-3, 5-13, 16-23, 25-28		
Xml	1-3, 5-7, 9-11, 26		