

The Switchboard Manager in MS-Access is a handy tool that provides for quick-and-dirty menu creation to organize the work flow. To help users visualize & understand the navigation, I created a form with a Treeview control, call it Treeview1.

When you review the following code, keep in mind that I have an item "GO BACK" on each switchboard page that takes the user back to the previous page.

```
Private Sub PopulateTreeview()
    Dim db As DAO.Database
    Dim key As String
    Dim nde As Node
    Dim rs1 As DAO.Recordset
    Dim rs2 As DAO.Recordset
    Dim rs3 As DAO.Recordset
    Dim s As String
    '
    Set db = CurrentDb
    s = "SELECT * FROM [Switchboard Items] " & _
        "WHERE SwitchboardID = 1 AND ItemNumber > 0 "
    Set rs1 = db.OpenRecordset(s)
    With Me.TreeView1
        If Not (rs1.BOF And rs1.EOF) Then
            rs1.MoveFirst
            ' create the topmost node
            Set nde = .Nodes.Add(, tvwAutomatic, "A", "Main Switchboard")
            Do While Not rs1.EOF
                ' add child nodes
                key = "B" & CStr(rs1!ItemNumber)
                Set nde = .Nodes.Add("A", tvwChild, key, rs1!ItemText)
                ' look for child nodes, off of this child
                s = "SELECT * FROM [Switchboard Items] " & _
                    "WHERE SwitchboardID = " & rs1!Argument & _
                    " AND ItemNumber > 0 " & _
                    " AND ItemText <> 'GO BACK' "
                Set rs2 = db.OpenRecordset(s)
                If Not (rs2.BOF And rs2.EOF) Then
                    rs2.MoveFirst
                    Do While Not rs2.EOF
                        key = "B" & CStr(rs1!ItemNumber) & "C" & CStr(rs2!ItemNumber)
                        Set nde = .Nodes.Add("B" & CStr(rs1!ItemNumber), tvwChild, key, rs2!ItemText)
                        ' look for child nodes, off of this child
                        s = "SELECT * FROM [Switchboard Items] " & _
                            "WHERE SwitchboardID = " & Val(rs2!Argument) & _
                            " AND ItemNumber > 0 " & _
                            " AND ItemText <> 'GO BACK' "
                        Set rs3 = db.OpenRecordset(s)
                        If Not (rs3.BOF And rs3.EOF) Then
                            rs3.MoveFirst
                            Do While Not rs3.EOF
                                key = "B" & CStr(rs1!ItemNumber) & _
                                    "C" & CStr(rs2!ItemNumber) & _
                                    "D" & CStr(rs3!ItemNumber)
                                Set nde = .Nodes.Add("B" & CStr(rs1!ItemNumber) & _
                                    "C" & CStr(rs2!ItemNumber), tvwChild, key, rs3!ItemText)
                                rs3.MoveNext
                            Loop
                        End If
                        rs3.Close
                        Set rs3 = Nothing
                        rs2.MoveNext
                    Loop
                End If
                rs2.Close
                Set rs2 = Nothing
                rs1.MoveNext
            Loop
        End If
    End With
    rs1.Close
    Set rs1 = Nothing
    Set db = Nothing
End Sub

Private Sub Form_Load()
    PopulateTreeview
End Sub
```

Simply create a form, add a Treeview control and invoke the code from the Form_Load event. I also add a Form_Activate event to DoCmd.Restore so the helper form always displays in a reduced window.