Information Box
Jeffrey Eldred

Overview:
The following is an excerpt from “Help and Information about this Control Panel”
This is a worksheet designed to store, organize, and toggle an
information system about this workbook. When 'Information Box' is
activated, a right-click on any cell in the workbook will search the
information stored in the top left corner of this worksheet for range that
includes that cell, and deliver a message box containing any relevant
information there. If not added manually, new information can be
added by activating the 'Information Adder' and then right-clicking
with a cell or range of cells selected. A prompt will then ask for the
information to be associated with this range. Finally clicking 'Remove
Information' will remove a row or rows of help information and adjust
the remaining rows to account for the loss. As of July 2007, real
information is yet to be added.

Detailed Operation and Structure:
Note: the simple content (“X”, “7”, and “@”) in the cells used in the examples below is
used for clarification purposes only. The information listed is not limited to or based on
the actual content of the cells that the information is associated with. The assumption is
that a correct analysis of the content would be provided to the information system which
would in turn be relayed back.
**Information Box (Help):** Provided that the Information Box is activated (Help.Value = True), and that only one cell is selected (Target), a simple search through the information list (the length of which is assumed but not asserted to be the value at “F1”) is conducted, and the function *Figure* is called to determine which ranges the selected cell is under and which corresponding pieces of information to add to the message. A cell with does not have pieces of information associated with it, obviously does not receive a message box.

```
Function Figure(ByVal sing As String, ByVal multi As String)
    If Help.Value Then
        Dim InfoList As String
        InfoList = ActiveSheet.ListObject.ListObject.List(1).Values(1)
        For Each Info In InfoList
            Dim InfoArray() As String
            InfoArray = Split(Info, vbCrLf)
            Dim InfoFound As Boolean
            For Each Line In InfoArray
                Dim InfoSections() As String
                InfoSections = Split(Line, vbCrLf)
                Dim SingSection As String
                Dim MultiSection As String
                For Each Section In InfoSections
                    Dim Sections() As String
                    Sections = Split(Section, vbCrLf)
                    Dim Sing As String
                    Dim Multi As String
                    Sing = Sections(1)
                    Multi = Sections(2)
                    If Sections(0) = sing Then
                        InfoFound = True
                        Exit For
                    End If
                    If Sections(0) = Multi Then
                        InfoFound = False
                        Exit For
                    End If
                Next Section
            Next InfoSection
            If InfoFound Then
                Help.Value = False
                Exit Function
            End If
        Next Info
    End If
    Help.Value = False
End Function
```

*Figure(ByVal sing As String, ByVal multi As String):* This function, called by *Help*, returns *True* if the cell at address *sing* is within the range at addresses *multi*. To do this, it delimits *multi* by commas and expresses each part as either a singular address or a range containing ‘:’ for which to loop through (the address reference style R1C1 is useful for this purpose). If at any point these singular addresses that make up *multi* are equivalent to *sing*, it returns *True*. If at no point it does, then it must be *False*. 
**Information Adder (Adder):** As both the Information Box and Information Adder trigger on a right-click when activated, neither can be activated at the same time as the other. The userform, InfoAdder, is shown and presents a textbox to input information to be associated with the selected cells. Which submit button is clicked determines the value of InfoAdder.formResult.Caption which verifies this information was intended to be used. If it was, then this information, and the associated range is added to the end of the list and the Help Size is incremented by one to accommodate. The corresponding entry for the screenshot below would read “L22,K16:M16,L17:M17,N19:O20” at A4, “information” at B4, and ‘@’ is a symbol representing "at"!’ at C4.

**Remove Information (Remove):** The userform, RHelp, is shown and presents two textboxes to input the range of rows (by actual row number, not by order within the list) to be removed. A check that this is valid input (numerical, within range, with the higher constraint above the lower) and which submit button is clicked determines the value of RHelp.formResult.Caption, which permits the process to continue. If it does, these rows are removed and the list and Help Size is adjusted to account for the loss. The result of the screenshot below would eliminate the first two information entries (about “X” and “7”) stored in rows 2 and 3, leaving only the recently added entry (about “@”) remaining, but in row 2 and with the value at cell F1 lowered to 1.
Possible Future Improvements:

- The most obvious improvement needed is the addition of information to this system. As long as it is empty it remains of no aid in any of the applications described in the overview.

- While the Information Box system provides unlimited information about a given cell, it is unable to provide information about graphs or drawings, which are an equally vital portion of any workbook. A proposed solution would be a similar or an added on information system that gives information about an entire worksheet, which would include graphs and drawings, as well as the overall structure of that worksheet. A combobox updated with a list of all worksheets (possibly used with or in a similar manner to ComboSheet, UpdateSheet, and VerifySh from the Quick Import Utility (complete technical note here and source code here)) would probably be the easiest method to refer to a given worksheet.

- The Information Box ignores any selection that includes anything more than one cell. However, the code could be modified so that information relevant to any of the cells is displayed, along with the cells that the information applies to. If this became too encumbering for a simple message box, then a textbox in the information worksheet would be easy to implement.