SubList Library - lbjoseph

The SubList library is a group of subs and functions that make it easy to store data inside a single string. Obviously, this isn't as fast an array, but it's an extremely easy way to create key-value pairs, commonly referred to as lists, hash tables, and dictionaries.

The Code

This piece of code needs to be at the top of your file:

```
' ------------------------------------ sl SubList Library -----------
------------------------- '
Global sl.d$ : sl.d$ = Chr$(1) ' Delimiter for data storage in sl.
' ------------------------------------------------------------------
------------------------- '
```

These subs and functions should be at the bottom of your file:

Table of Contents


```
' ------------------------------------ sl SubList Library -----------
------------------------- '
```

```
' ------------------------------------------------------------------
------------------------- '
Sub sl.Set BYREF list$, key$, content$
    ' Adds the content$ to the list$ with the index key$.
    found = Instr(list$, sl.d$;key$;sl.d$;sl.d$)
    If found Then
        ' Key is already found in list:
        contentStart = found + Len(key$) + 3
        contentEnd = Instr(list$, sl.d$, contentStart)
        If Not(contentEnd) Then contentEnd = Len(list$)+1
        contentLength = contentEnd - contentStart
        halve1$ = Mid$(list$, 0, contentStart)
        halve2$ = Mid$(list$, contentEnd)
        list$ = halve1$; content$; halve2$
    Else
        ' Key needs to be added to list.
        list$ = list$; sl.d$;key$;sl.d$;sl.d$;content$
    End If
End Sub
Function sl.Get$(list$, key$)
    ' Adds the content$ to the list$ with the index key$.
    found = Instr(list$, sl.d$;key$;sl.d$;sl.d$)
    If found Then
        ' Key is already found in list:
        contentStart = found + Len(key$) + 3
        contentEnd = Instr(list$, sl.d$, contentStart)
        If Not(contentEnd) Then contentEnd = Len(list$)+1
        contentLength = contentEnd - contentStart
        content$ = Mid$(list$, contentStart, contentLength)
        sl.Get$ = content$
    End If
End Function
Sub sl.Remove BYREF list$, key$
    ' Adds the content$ to the list$ with the index key$.
    found = Instr(list$, sl.d$;key$;sl.d$;sl.d$)
    If found Then
        ' Key is already found in list:
        contentStart = found + Len(key$) + 3
        contentEnd = Instr(list$, sl.d$, contentStart)
        If Not(contentEnd) Then contentEnd = Len(list$)+1
        contentLength = contentEnd - contentStart
        halve1$ = Mid$(list$, 0, found)
        halve2$ = Mid$(list$, contentEnd)
        list$ = halve1$; content$; halve2$
    End If
End Sub
```

```
Function sl.Keys(list$, BYREF keys$)
    ' Returns the number of keys in list$ (n).
    ' The keys are indexed 1 through n in the provided list keys$.
    w = 1
    del$ = sl.d$;sl.d$
    While Word$(list$, w, del$) <> ""
        value$ = Word$(list$, w, del$)
        If w = 1 Then
            keys = 1
            k$ = Mid$(value$,2)
            Call sl.Set keys$, Str$(keys), k$
        Else
            k$ = Word$(value$, 2, sl.d$)
            If k$ <> "" Then
                keys = keys + 1
                Call sl.Set keys$, Str$(keys), k$
            End If
        End If
        w = w + 1
    WEnd
    sl.Keys = keys
End Function
' ------------------------------------------------------------------------
------------------------- '
' ------------------------------------------------------------------------
------------------------- '
```

Using the SubList Library

The following sections detail the use of the SubList library. In the SubList library, each piece of data (or value) has an index that allows it to be easily accessed. This index can be more than just a number. It can be any word or string, as long as it doesn't contain the character Chr$(1). Indexes are called keys. For example, you might have a key called "person 1", and the corresponding value for that might be "Bob Williams".

Creating a List

To create a new list to store data, one simply initializes a string variable:

```
myList$ = ""
```

Adding Data

To store data inside the list, one just needs to call the sl.Set subroutine and specify the list, the key name, and the value for that key.

```
Call sl.Set myList$, "key a", "The letter 'a'."
```

Retrieving Data

To get the value of a specified key in the list, one must be call the sl.Get$() function.

```
value$ = sl.Get$(myList$, "key a")
```

In this case, value$ would be equal to "The letter 'a'."

Simply enough? The author thought so.

Getting a List of Keys

Sometimes one would like to get a list of all the keys in a list. This can be useful if one sets a bunch of values in a list with different kinds of keys, and wants to retrieve every value from the list.

Fortunately, the SubList library has a function to return the number of keys in a list, and a list of those keys with numeric indexes. Here's a quick example:

```
items = 7    ' Number of items to make.
myList$ = "" ' List with values.
keys$ = ""   ' List to hold the keys in myList$.
keys = 0     ' Number of keys in myList$.
For i = 1 To items
    Call sl.Set myList$, "key ";Chr$(96+i), "Entry ";i
Next i
' Get the list of keys now:
keys = sl.Keys(myList$, keys$)
For i = 1 To keys
    keyName$ = sl.Get$(keys$, Str$(i))
    Print "  Key ";i;":", keyName$, "Data: ", sl.Get$(myList$,
 keyName$)
Next i
```