Chapter 36: Scripting.Dictionary object

You must add Microsoft Scripting Runtime to the VBA project through the VBE's Tools → References command in order to implement early binding of the Scripting Dictionary object. This library reference is carried with the project; it does not have to be re-referenced when the VBA project is distributed and run on another computer.

Section 36.1: Properties and Methods

A Scripting Dictionary object stores information in Key/Item pairs. The Keys must be unique and not an array but the associated Items can be repeated (their uniqueness is held by the companion Key) and can be of any type of variant or object.

A dictionary can be thought of as a two field in-memory database with a primary unique index on the first 'field' (the *Key*). This unique index on the Keys property allows very fast 'lookups' to retrieve a Key's associated Item value.

Properties

name	read/write	type	description
CompareMode	*read / write*	CompareMode constant	Setting the CompareMode can only be performed on an empty dictionary. Accepted values are 0 (vbBinaryCompare), 1 (vbTextCompare), 2 (vbDatabaseCompare).
Count	*read only*	unsigned long integer	A one-based count of the key/item pairs in the scripting dictionary object.
Key	*read / write*	non-array variant	Each individual unique key in the dictionary.
Item(*Key*)	*read / write*	any variant	Default property. Each individual item associated with a key in the dictionary. Note that attempting to retrieve an item with a key that does not exist in the dictionary will *implicitly add* the passed key.

Methods

name	description
Add(*Key*,*Item*)	Adds a new Key and Item to the dictionary. The new key must not exist in the dictionary's current Keys collection but an item can be repeated among many unique keys.
Exists(*Key*)	Boolean test to determine if a Key already exists in the dictionary.
Keys	Returns the array or collection of unique keys.
Items	Returns the array or collection of associated items.
Remove(*Key*)	Removes an individual dictionary key and its associated item.
RemoveAll	Clears all of a dictionary object's keys and items.

Sample Code

```
'Populate, enumerate, locate and remove entries in a dictionary that was created
'with late binding
Sub iterateDictionaryLate()
    Dim k As Variant, dict As Object

    Set dict = CreateObject("Scripting.Dictionary")
    dict.CompareMode = vbTextCompare          'non-case sensitive compare model

    'populate the dictionary
    dict.Add Key:="Red", Item:="Balloon"
    dict.Add Key:="Green", Item:="Balloon"
    dict.Add Key:="Blue", Item:="Balloon"
    'iterate through the keys
    For Each k In dict.Keys
        Debug.Print k & " - " & dict.Item(k)
    Next k

    'locate the Item for Green
    Debug.Print dict.Item("Green")

    'remove key/item pairs from the dictionary
    dict.Remove "blue"      'remove individual key/item pair by key
    dict.RemoveAll          'remove all remaining key/item pairs

End Sub

'Populate, enumerate, locate and remove entries in a dictionary that was created
'with early binding (see Remarks)
Sub iterateDictionaryEarly()
    Dim d As Long, k As Variant
    Dim dict As New Scripting.Dictionary

    dict.CompareMode = vbTextCompare          'non-case sensitive compare model

    'populate the dictionary
    dict.Add Key:="Red", Item:="Balloon"
    dict.Add Key:="Green", Item:="Balloon"
    dict.Add Key:="Blue", Item:="Balloon"
    dict.Add Key:="White", Item:="Balloon"

    'iterate through the keys
    For Each k In dict.Keys
        Debug.Print k & " - " & dict.Item(k)
    Next k

    'iterate through the keys by the count
    For d = 0 To dict.Count - 1
        Debug.Print dict.Keys(d) & " - " & dict.Items(d)
    Next d

    'iterate through the keys by the boundaries of the keys collection
    For d = LBound(dict.Keys) To UBound(dict.Keys)
        Debug.Print dict.Keys(d) & " - " & dict.Items(d)
    Next d

    'locate the Item for Green
    Debug.Print dict.Item("Green")
    'locate the Item for the first key
    Debug.Print dict.Item(dict.Keys(0))
    'locate the Item for the last key
    Debug.Print dict.Item(dict.Keys(UBound(dict.Keys)))

    'remove key/item pairs from the dictionary
    dict.Remove "blue"                         'remove individual key/item pair by key
    dict.Remove dict.Keys(0)                   'remove first key/item by index position
    dict.Remove dict.Keys(UBound(dict.Keys))   'remove last key/item by index position
    dict.RemoveAll                             'remove all remaining key/item pairs

End Sub
```