# Review Text Files

In this modern world of ours, a new type of English seems to have evolved – Texting English. Just one example is the abbreviation LOL, the meaning of which is “Laughing out loud”. Your project for review number seven is to write a program that will serve as a Texting English/Real English dictionary. Here’s how Part One of your program will work (the review is in two parts).

##### Part One

There is a text file to download called “dictionary”. This contains Texting English followed by its Real English translation. For example:

##### LOL = Laughing out loud

Each line uses an equals sign to separate the Texting English from its Real English translation. When a button is clicked on your form, you job is to write the code that opens this “dictionary” text file and read its contents. Once the text file is opened, write code to place the Texting English in one List Box and the Real English in a second List Box.

That’s clearly not enough, though. When a user clicks on a term in a List Box, display the translation in a Textbox. So if LOL was selected from your List Box, the translation “Laughing out loud” should appear in the textbox. But if “Laughing out loud” was selected, the Texting version, LOL, should appear in the textbox.

##### Part Two

Add two new textboxes and a button. Write code so that the user can add a new Text/Real English pair to the dictionary.

##### Help for this Review

**Part One Help**

When you create your form, you’re going to need a Button and two List boxes. When the button is clicked, you first have to open the text file called **dictionary**. We’ll assume that this text file is in the root folder of C. So the file path would be:

##### TextFile = “C:\dictionary.txt”

Once you have opened the file, you need to read it line by line (you can use a Do While loop for this). Each line will have two parts: the Texting English and the Real English:

##### LOL = Laughing out loud

What you have to do is to Split each line in two. The line separator is the equals sign. You can then use the Split method of a string variable to search for this separator. Because each line has only two parts, you can set up an array with two elements in it:

##### Dim AryLine(1) As String

When you use Split, you can put the first part into array position zero, and the second part into array position 1.

Once you have split the line in two, you can then add each part to the list boxes. The way to add items to a list box is like this, remember:

##### Listbox1.Items.Add()

In between the round brackets, you put what it is you want to add. In your case, you’ll want to put something from your array:

##### Listbox1.Items.Add( AryLine(0) )

When you’ve finished looping round the text file, you’ll then have two list boxes filled with the dictionary pairs. Once you have that working correctly, you can move on to your next objective

* selecting an item from a List Box.

##### How to Select an item from a List Box

When you add an item to a list box, you’re creating a collection. The first item in the collection is placed at position zero, the next at position 1, the next at position 2, etc. You can use the index position of the collection to get at the items. The property of the List Box that does this is the **SelectedIndex** property. Here’s some code that’s been added to the Click event of a List box with the Name of **lstText**:

##### Private Sub lstText\_Click(sender As Object, e As EventArgs) \_

**Handles lstText.Click**

**Dim selIndex As Integer**

**selIndex = lstText.SelectedIndex MessageBox.Show( selIndex )**

**End Sub**

The **SelectedIndex** will be a number. This number is the position of the item in the list Box. If you were to run that code, then click on an item in the list box, a number would be displayed. If you clicked on the first item, this number would be 0; if you clicked on the third item, this number would be 2 (the count starts from zero, remember).

You can use this number to get at the Text property of the item – in other words, what the item is that was selected. The way you use the number is like this:

##### ItemChosen = lstText.Items.Item(selIndex)

This somewhat unintuitive line says “There is a property of the List Box called Items. This Items property has a property of its own called Item. Access this Item property.”

But the Item property needs a number between its round brackets. The number corresponds to its position in the List Box. This will then return the text of the item chosen.

(HINT: The item position in each of your two list Boxes will be the same. So if LOL is at position 2 in your first list box, “Laughing out loud” will be at position 2 in your second list box. So if a user selects item 2 from list box one, you can grab item 2 from list box two. This will then be the translation you need)

Once you have the translation you need, you can place it into your textbox. Part One will then be completed.

##### Help with Part Two

For part two, you have to allow a user of your program to add a new entry to the dictionary. The Text English will be entered into one textbox, and the Real English entered into the other. When a button is clicked, the item is added to the dictionary.

Part two should be a lot easier. All you’re doing here is taking text from two textboxes and joining it together, along with an equals sign in the middle. Once you’ve joined the text, you can then append the text to your dictionary text file.

You should test for errors. What if a user leaves one textbox blank? What if your text file has been moved or deleted? Make sure you code for this.