
VALIDATION OBJECT WITH VOICE FEEDBACK

Speech-Enhanced Microsoft Excel (XP/2003/2007) Project

Dr. Alexander Bell

DISCLAIMER OF WARRANTY AND LIMITATION OF LIABILITY

This article and all related products are provided on AS IS basis without warranty of any kind for Demo/Evaluation purpose only. Commercial use of this product is strictly prohibited. In no case the Author and/or his Company could be held liable for any damages related to this article and associated products/solutions.

INTRODUCTION

The main goal of this Demo project is to demonstrate the idea of building Speech-enhanced Excel application, extending the capability of its standard data Validation Object (VO) by adding Voice Feedback (VF) User notification. VF is based on so-called “Text To Speech” (TTS) speech synthesis technology, which comes with Microsoft Excel 2002/2003/2007

The whole solution is embedded into a single MS Excel file. The source code is rather small and transparent, easy to use and to learn, serving both practical and didactical purposes. The same principles and code components could be re-used in other speech-enabled business applications. Core components of the projects are: Data Validation Object (VO) and the Speech Object (SO), both included in Ms Excel 2002/2003. These objects are discussed in greater details in following chapters.

DATA VALIDATION OBJECT: CONCEPT AND IMPLEMENTATION

Data Validation is the crucial part of any well designed UI, thus it seems rather reasonable to have the dedicated Validation Object (VO) presented in MS Excel.

VO is a part of Excel Object Library. Its main goal is to ensure that the data entry complies with certain user-defined criteria set for the whole worksheet range or just a single cell; in simple words, VO serves as a data compliance watchdog. Normally, it communicates with the user through its visual components, namely Message Boxes, which are modal Windows Forms. To close the message box user has to perform additional operation. VF is introduced here as an alternative Audio notification to user.

Validation could be added manually (use Menu Item: DataàValidation) or programmatically. Adding VO to the data range manually is rather intuitive; there are a lot of similarities with Cell conditional formatting in Excel worksheet.

VO could be added programmatically by using the **Validation.Add** method. Further, you could apply the **Modify** method to the existing VO or use **Delete** method to remove it.

VO has several properties, which specifies the details of its behavior. To better understand the intrinsic logic of the VO, let's think for a moment about data validation on the conceptual level as of a two-state process.

- 1** The feature must prompt the User about validation rules before the actual data was entered. This first-state “preemptive” data validation procedure is triggered when User select the Cell.
- 2** Actual Data Validation event procedure must be triggered after the data entry is completed. If validation error is detected, VO should send to the User a warning message and (optionally) halt the program execution till the error is cleared.

This general data validation concept is implemented in the MS Excel VO. Any Worksheet could contain many VOs, associated with different Ranges.

There are several VO properties, associated with “preemptive” prompt, which pop-ups as the user select the Cell containing VO. These properties are:

- ShowInput,**
- InputTitle,**
- InputMessage.**

Text strings, assigned to the InputTitle and InputMessage properties are displayed as the Title and the Body of the prompt, respectively (see Figure 1). This prompt appears automatically on the screen when Cell containing VO is selected. User could turn this prompt On/Off by setting the ShowInput property to True/False, respectively. In our project we will keep this feature On.

Item #	Task Description	Start Date	End Date	Labor, hrs
1	AVUI: Voice Command Interface	6/1/2002	6/19/2002	104
1.1	General Concept	6/1/2002	6/5/2002	24
1.2	Coding, Testing and Debugging	6/6/2002	6/10/2002	
1.3	Documenting: Text	6/12/2002	6/17/2002	
1.4	Documenting: Graphics	6/18/2002	6/18/2002	
1.5	Final Package checklist and delivery	6/19/2002	6/19/2002	
2	AVUI: Validation Object with Voice Feedback	6/20/2002	6/26/2002	10
2.1	General Concept	6/20/2002	6/20/2002	
2.2	Coding, Testing and Debugging	6/21/2002	6/23/2002	
2.3	Documenting: Text	6/24/2002	6/24/2002	
2.4	Documenting: Graphics	6/25/2002	6/25/2002	
2.5	Final Package checklist and delivery	6/26/2002	6/26/2002	8

Figure 1. This Excel-based timetable and labor calculator demonstrates the use of speech enhanced Validation Object (VO) combined in business applications. Validation rules, added programmatically to the Range ("C6:D10, C13:D17"), allow only the Dates between 01/01/2002 and 12/31/2002 inclusively to be entered. Column E contains NetWorkdays() worksheet function for labor calculation

Another set of properties relates to the “Post Factum” state of the validation process:

- ShowError,**
- ErrorTitle,**
- ErrorMessage,**

If the property **ShowError** was set to True and the validation rules were violated, then the Validation Error Message Box will pop-up, containing ErrorTitle and ErrorMessage text. Depends on the AlertStyle property, which value could be set to either xValidAlertInformation, xValidAlertStop or xValidAlertWarning, the Message Box will appear either as a 3 buttons “YES-NO-Cancel” (see Figure 2), or 2 buttons “Retry-Cancel” (see Figure 3) style.

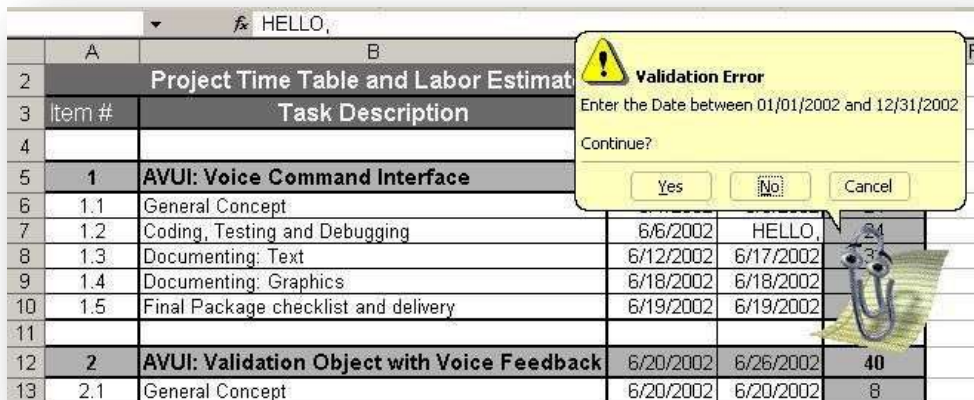


Figure 2. Validation Object (VO) Error Message: VO is reacting to the invalid data entry “HELLO” in the Cell “D7”, where only type Date is allowed. Message Box contains YES-NO-CANCEL Buttons: AlertStyle set to xValidAlertWarning

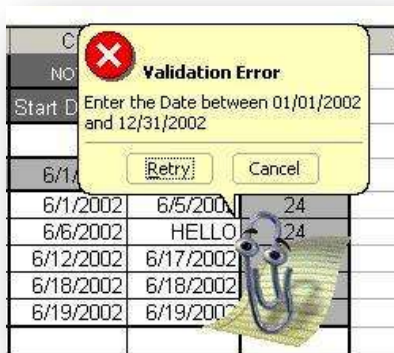


Figure 3. VO Error Message: Invalid data entry detected in Cell “D7” causes VO to pop-up the 2-Buttons style Error Message Box of “RETRY-CANCEL” (AlertStyle set to xValidAlertStop)

Validation rules can be set programmatically by using the Operator, Formula1 and Formula2 properties of the VO, which are similar to that used in Conditional Formatting (for more details and samples, please consult to the MS Help built-in feature on Validation).

Sample procedure to add the VO to the data entry area, namely: Range ("C6:D10, C13:D17") is shown below (see the code Listing 1). This procedure is called by **Workbook_Open** event procedure. Notice, that adding the VO to the Range Object should be done only once: the second and any subsequent attempt to add VO to the same Range will cause the trappable error to happen and the procedure will be terminated, causing no harm.

Listing 1. Adding Validation Object (VO) to MS Excel Range programmatically.

```
Private Sub AddValidation()
```

```
On Error GoTo ErrorHandler
```

```
    With Worksheets(1).Range("C6:D10, C13:D17").Validation
```

```
        .Add Type:=xlValidAlertWarning, _
```

```
        AlertStyle:=xlValidAlertWarning, _
```

```
        Operator:=xlBetween, _
```

```
        Formula1:="01/01/2002", _
```

```
        Formula2:="12/31/2002"
```

```
        .InputTitle = "Validation Rule"
```

```
        .InputMessage = "01/01/2002 <= Date <= 12/31/2002"
```

```
        .ShowInput = True
```

```
        .ErrorTitle = "Validation Error"
```

```
        .ErrorMessage = _
```

```
        "Enter the Date between 01/01/2002 and 12/31/2002"
```

```
        .ShowError = False
```

```
    End With
```

```
Exit Sub
```

```
ErrorHandle:
```

```
End Sub
```

```
Private Sub Workbook_Open()
```

```
    Call AddValidation
```

```
End Sub
```

VOICE FEEDBACK (VF) AS AN ALTERNATIVE USER NOTIFICATION METHOD

Speech Object (SO) enables to play back the text passed as a string argument (it also enables reading any Cell contents, but this feature is not in the scope of current project). This exciting feature was added to Excel starting from XP version (2002). Surprisingly I did not find it in other

Office XP/2003 applications, but it is easily accessible by referencing the Excel Object Library. Combined with VO discussed above it will extend the Man-Machine interaction principles by adding speech component to it. It could be achieved in 3 simple steps:

1 Suppress the standard Validation Error message. Setting the **ShowError** property of the VO to False will suppress the Validation Error Message: it will not appear on the screen even if the system detects such an error. Instead, the Voice Feedback will be used as a substitute for the Error Message Box, saving the User at least one mouse click per error.

2 Add the line of code, forcing the application to return focus to the Cell, which caused the validation problem and clear its content. To implement this feature there are at least two Events to choose from:

Workbook_SheetChange (ByVal Sh As Object, ByVal Target As Range)

Worksheet_Change (ByVal Target As Range)

You could use either of them up to your personal preferences. For the consistency, the first one was chosen, thus all project code resides in the single ThisWorkbook code module.

3 Use **Speech Object (SO)** to read the Validation Error text. As it was mentioned above, SO allows to playback the text string, passed as an argument to its Speak method. The general Syntax for SO Speak method is as following:

Application.Speech.Speak(Text, SpeakAsync, SpeakXML, Purge),

where **Text** (the only required parameter) represents the string to be spoken. Other parameters are optional and are explained in the Table 1:

Table 1. Optional parameters of Speech.Speak method

Parameter	Default Value	Description
SpeakAsync	False	True will cause the Text to be spoken asynchronously and vice versa
SpeakXM	False	True will cause the Text to be interpreted as an XML and vice versa
Purge	False	True will cause the current speech to be terminated and any buffered text to be purged

Final subroutine utilizes Speech.Speak method in asynchronous mode to enable the VF in VO (see code Listing 2). It works as follows: when the data in any Cell changes, the system will check for the validation error using Target.Validation.Value property. In case this property value is set to False, which indicates the validation error, the Cell's content will be cleared and VF will be activated, reading to the User the content of the ErrorTitle and ErrorMessage of the suppressed validation error Message Box.

Listing 2. Voice Feedback serves as a substitute for the VO's validation error message: it reads the ErrorTitle and ErrorMessage text, using the MS Excel Application.Speech Object.

```
Private Sub Workbook_SheetChange(ByVal Sh As Object, ByVal Target As Range)  
On Error Resume Next  
    With Target  
        If Not (.Validation.Value) Then  
            .ClearContents  
            .Activate  
            Application.Speech.Speak _  
            (.Validation.ErrorTitle & ". " & _  
            .Validation.ErrorMessage), True  
        End If  
    End With  
End Sub
```

TEXT-TO-SPEECH FINE TUNING

Couple more things could be done with Text-To-Speech feature. Go to “ControlPanel-> Speech -> Text To Speech” Menu sub-item. It will open the multi-tabbed dialog screen, looking like the one shown on Figure 4. From this dialog you could choose the voice patterns (either LH Michael or LH Michelle) and set the desirable speed.



Figure 4. Text To Speech Dialog Screen: choose the Voice speed and pattern up to your personal preferences.

CREATING EXCEL TEMPLATE FILE

To finalize the project, place the code snippets from Listing 1 and 2 into **ThisWorkbook** code module, compile it and close the VBE window. Then format the first Worksheet in a Workbook as it's shown on Figure 1 (you could add your own task description) and save the file as a Template with .xlt extension in your MS Excel Template Directory; it is ready for use. To open MS Excel file (the .xls one) based on this “Project Timetable” Template, go to “File ->New” Menu item and choose the Template file you've just created.

REFERENCES

1. Alexander Bell. Build a Validation Object with Voice Feedback, ACCESS-VB-SQL ADVISOR, Dec, 2002
2. Alexander Bell. Excel at Audio-Visual UI Development, ACCESS-VB-SQL ADVISOR, Nov, 2002