Metodi e Proprietà del Foglio di lavoro o Worksheet

L'oggetto ***foglio*** rappresenta un singolo foglio in una cartella e si riferisce a una raccolta di tutti i fogli in una cartella di lavoro, cioè tutti i fogli di lavoro, fogli grafici, macro etc. ed è un membro sia della collezione Worksheets (oggetto foglio di lavoro) che della raccolta Sheets (oggetto Foglio). La proprietà ***Workbook.Worksheets*** restituisce un insieme Worksheets (cioè un oggetto foglio), che si riferisce a tutti i fogli di una cartella di lavoro, mentre la proprietà *Workbook.Sheets* restituisce un insieme Sheets, che si riferisce a tutti i fogli di una cartella di lavoro. Utilizzando il codice *MsgBox ActiveWorkbook.Worksheets.Count* restituirà il numero dei fogli di lavoro nella cartella di lavoro attiva, e il codice *MsgBox ActiveWorkbook.Sheets.Count* restituirà il numero dei fogli nella cartella di lavoro attiva.

L'oggetto *Charts* (grafico) rappresenta un diagramma in una cartella di lavoro, che può essere sia un grafico incorporato o un foglio grafico separato e la collezione Charts si riferisce ad una raccolta di tutti i fogli grafici in una cartella, ed esclude tutti i grafici incorporati, mentre invece l'oggetto *ChartObject* rappresenta un grafico incorporato e si riferisce ad una raccolta di tutti gli oggetti ChartObject in un unico foglio, cioè in un foglio grafico specifico.

Una cartella di lavoro può contenere quattro tipi di fogli, il foglio di lavoro, un foglio con grafico, fogli macro (Macro MS Excel 4.0) e un foglio di dialogo (finestra di dialogo MS Excel 5.0). I fogli macro (chiamati anche macro XLM) e i fogli di dialogo (utilizzati nelle versioni precedenti di Excel per creare finestre di dialogo personalizzate, ora sostituiti dalle UserForm), sono ancora forniti e supportati in Excel 2007 solo per compatibilità con le versioni precedenti di Microsoft Excel, mentre un foglio macro (o un foglio di dialogo) non è incluso come parte della collezione dei fogli di lavoro, ma è una parte della collezione Sheets.

Esempio: Illustrare i tipi di oggetti spiegati sopra, considerando una cartella di lavoro con tre fogli di lavoro denominati: Foglio1, Foglio2 e Foglio3), 2 grafici incorporati in Foglio1, 1 foglio grafico Chart1.

```
Sub prova1()
Dim ws As Worksheet, i As Integer
MsgBox ThisWorkbook.Sheets.Count 'Restituisce 6, in quanto ci sono: 3 fogli
For i = 1 To ThisWorkbook.Sheets.Count 'Restituisce i nomi di ognuno dei 6 fogli
MsgBox ThisWorkbook.Sheets(i).Name
Next i
MsgBox ThisWorkbook.Worksheets.Count 'Restituisce 3 (che sono i fogli di lavoro)
For Each ws In ThisWorkbook.Worksheets 'Restituisce i nomi di ciascuno dei 3 fogli di lavoro
MsgBox ws.Name
Next
Dim sh As Object
For Each sh In ThisWorkbook.Sheets 'Restituisce i nomi di ognuno dei 6 fogli
MsgBox sh.Name
Next
For Each sh In ThisWorkbook.Worksheets 'Restituisce i nomi di ognuno dei 3 fogli di lavoro
MsgBox sh.Name
Next
MsgBox ThisWorkbook.Charts.Count 'restituisce 1 - c'è 1 oggetto grafico "Chart1"
MsgBox Sheets("Chart1").ChartObjects.Count 'restituisce 0 - non c'è grafico nel foglio
'restituisce 2 - ci sono 2 grafici incorporati ChartObject nel foglio di lavoro "Foglio1"
MsgBox Sheets("Sheet1").ChartObjects.Count
End Sub
```

Riferimenti a un singolo foglio di lavoro

Per fare riferimento o restituire un oggetto foglio di lavoro (unico foglio di lavoro), usiamo le proprietà di entrambi i fogli di lavoro e l'oggetto foglio di lavoro, come illustrato di seguito. La proprietà Item dell'oggetto foglio; es. *Worksheets.Item* si riferisce ad un singolo foglio in una collezione e presenta la seguente sintassi: *WorksheetsObject.Item (Index)*, dove *Index* è il nome del foglio di lavoro o il numero di indice, che però è anche possibile omettere utilizzando il 'punto' in una sintassi come:

```
WorksheetsObject (WorksheetName)
'oppure
WorksheetsObject (IndexNumber)
```

Il numero di indice inizia da 1, viene visualizzato per primo ed è collocato all'estrema sinistra del foglio di lavoro nella barra delle schede della cartella di lavoro che si incrementa verso destra per ogni foglio presente, inclusi i fogli di lavoro nascosti, e l'ultimo foglio collocato all'estrema destra del foglio di lavoro viene restituito da *Worksheets (Worksheets.Count)* . La proprietà *Count* dell'oggetto fogli di lavoro restituisce il numero di fogli nella collezione (cioè nella cartella di lavoro), inoltre per riferirsi a un singolo foglio nella raccolta fogli è possibile utilizzare la proprietà ***Sheets.Item*** con la seguente sintassi:*SheetsObject.Item (Index)*

La scheda del foglio di lavoro visualizza il nome del foglio stesso e si utilizza la proprietà ***Name*** dell'oggetto Worksheet per impostare o restituire il nome di un foglio di lavoro usando questa sintassi: *WorksheetObject.Name*. Di seguito alcuni esempi di codice con riferimento a un foglio di lavoro

Restituisce il nome del primo foglio nella cartella attiva, sia usando o omettendo l'istruzione "Item"

```
MsgBox ActiveWorkbook.Worksheets.Item (1). Name
'oppure
MsgBox ActiveWorkbook.Worksheets (1). Name
```

Restituisce il nome del primo foglio della cartella di attiva, sia usando o omettendo l'istruzione "Item"

```
MsgBox ActiveWorkbook.Sheets.Item (1). Name
'oppure
MsgBox ActiveWorkbook.Sheets (1). Name
```

Restituisce il nome del primo foglio ThisWorkbook, omettendo l'istruzione "Item"

```
MsgBox ThisWorkbook.Worksheets (1). Name
```

Attiva il foglio di lavoro denominato Foglio3 della cartella di lavoro attiva

```
ActiveWorkbook.Worksheets ("Foglio3"). Activate
```

Attiva il foglio denominato Chart1 della cartella di lavoro attiva

```
ActiveWorkbook.Sheets ("Chart1"). Activate
```

Non specificando una cartella di lavoro si farà riferimento alla cartella di lavoro attiva, si usa il seguente codice per restituire il nome del secondo foglio di lavoro nella cartella di lavoro attiva:

```
MsgBox Worksheets (2). Name
'Oppure
MsgBox Sheets(2). Name
```

Consultare il foglio attivo nella cartella di lavoro attivo

```
MsgBox ActiveSheet.Name
```

Restituisce il nome del secondo foglio di lavoro nella cartella di lavoro prova.xlsx

```
MsgBox Workbooks ("prova.xlsx"). Worksheets (2). Name
```

Restituisce il nome del primo foglio di lavoro nella seconda cartella di lavoro

```
MsgBox Workbooks(2).Worksheets(1).Name
MsgBox Workbooks.Item(2).Worksheets.Item(1).Name
MsgBox Workbooks(2).Worksheets.Item(1).Name
```

Esempio: Consultare i fogli di lavoro della cartella di lavoro attiva contenente 3 fogli, vale a dire Foglio1, Foglio2 e Foglio3

```
Sub prova2()
'Restituisce il nome del primo foglio di lavoro "Foglio1"
MsgBox ActiveWorkbook.Worksheets(1).Name
'Cambia il nome del primo foglio di lavoro da "Foglio1" a "pippo"
ActiveWorkbook.Worksheets(1).Name = "pippo"
'Restituisce il nuovo nome del primo foglio di lavoro (pippo)
MsgBox Worksheets(1).Name
'Restituisce il nome dell'ultimo foglio (Foglio3)
MsgBox ActiveWorkbook.Worksheets(Worksheets.Count).Name
'Restituisce 3, il n° totale dei fogli di lavoro
MsgBox ActiveWorkbook.Worksheets.Count
'Restituisce il valore -1, che indica che il foglio di lavoro "Foglio2" è visibile
MsgBox Worksheets("Sheet2").Visible
'Nasconde "Foglio2”
Worksheets("Sheet2").Visible = False
'Restituisce il valore 0, che indica che "Foglio2" è nascosto
MsgBox Worksheets("Sheet2").Visible
'Restituisce 3, il n° totale di fogli di lavoro contando anche quelli nascosti
MsgBox Worksheets.Count
'Rende visibile "Foglio2"
Worksheets("Sheet2").Visible = True
End Sub
```

Facendo riferimento a un intervallo di celle, omettendo il qualificatore oggetto – *worksheet object* - per impostazione predefinita assume *Active Sheet*, vale a dire che utilizzando il codice *Range ("A1")* restituirà la cella A1 del foglio attivo, e sarà lo stesso che utilizzare *Application.Range ("A1")* o*ActiveSheet.Range ("A1")*.

Esempio: Immettere il valore 10 nella cella A1 del foglio di lavoro attivo

```
ActiveSheet.Range ("A1"). Value = 10
Range ("A1"). Value = 10
Application.Range ("A1"). Value = 10
```

L'oggetto attivo o ActiveSheet

Se non viene specificata la cartella di lavoro, Excel si riferisce alla cartella di lavoro corrente o attiva di default e nel codice VBA è possibile anche consultare l'attuale cartella di lavoro attiva come *ActiveWorkbook* o ***ActiveSheet*** e entrambe le espressioni *Worksheets (1). Name* e *ActiveWorkbook.Worksheets (1). Name* restituiranno il nome del primo foglio di lavoro nella cartella di lavoro attiva che diventa anche l'oggetto di default in questo caso. Allo stesso modo, entrambe le espressioni *Range ("A1"). Value = 56* e *ActiveSheet.Range ("A1"). Value = 56* inseriranno il valore 56 nella cella A1 del foglio di lavoro attivo nella cartella di lavoro attiva. Questa è una regola generale che omettendo il riferimento di una cartella di lavoro o foglio di lavoro ci si riferisce alla cartella di lavoro attiva o di default, ma è soggetto alle seguenti condizioni:

- Quando il codice VBA viene inserito nei moduli foglio, omettere il riferimento ad un foglio di lavoro si farà riferimento al foglio specifico in cui il il codice viene inserito nel modulo e non al *foglio attivo*
- Quando il codice VBA viene inserito nel modulo *Workbook (ThisWorkbook)*, omettere il riferimento ad una cartella di lavoro si farà riferimento alla cartella di lavoro in cui è inserito il codice e **NON** alla *cartella di lavoro attiva*. Questo significa che, omettendo il riferimento ad un foglio di lavoro si imposterà *ActiveSheet* quando il codice VBA viene inserito nei moduli standard di codice (Module1, Module2, etc.) oppure nel modulo della cartella di lavoro (ThisWorkbook) e **NON** quando il codice VBA viene inserito nei moduli Foglio (Foglio1, Foglio2, etc.) o Form o eventuali moduli di classe
- Omettendo il riferimento a una cartella di lavoro, quella predefinita sarà *ActiveWorkbook* quando il codice VBA viene inserito nei moduli di codice standard (Module1, Module2, etc.) o nei moduli foglio (Foglio1, Foglio2, etc.) e **NON** quando il codice VBA viene inserito nel modulo *Workbook*(ThisWorkbook).

Codice Name e Sheet Name

Nel Progetto VBE, la cartella *Objects* è sempre presente e contiene un oggetto foglio per ogni foglio di lavoro esistente, e un oggetto ThisWorkbook. Ogni oggetto foglio ha come primo nome il nome in codice del foglio, che appare al di fuori delle parentesi, e come secondo nome, che compare dopo il nome in codice e tra parentesi, il nome della scheda del foglio che appare nel foglio di lavoro di Excel. Il nome in codice del foglio di lavoro selezionato viene visualizzato a destra del "nome" nella finestra Proprietà, mentre il nome del foglio viene visualizzata a destra del nome quando si scorre verso il basso nella finestra Proprietà. Per impostazione predefinita, quando si aggiunge un foglio di lavoro, il nome in codice e il nome del foglio sono gli stessi e i nomi in codice di default e i fogli di default iniziano da Foglio1, Foglio2 etc. in questo ordine da sinistra a destra.

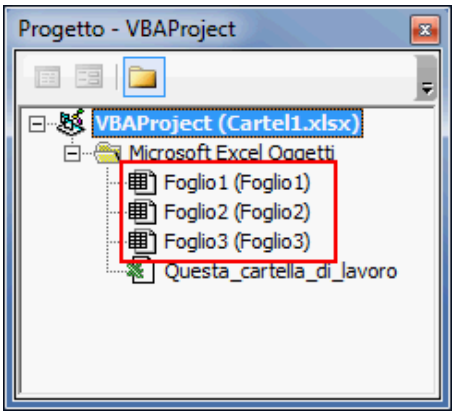

Fig. 1

È possibile modificare sia il nome in codice che il nome del foglio, mentre il nome del foglio può essere modificato nella finestra Proprietà (in VBE) o nella scheda foglio o da codice VBA, ma il nome in codice può essere modificato solo nella finestra Proprietà e non a livello di programmazione. Sia il nome in codice che il nome del foglio possono essere utilizzati durante la scrittura del codice. Mostriamo di seguito come utilizzare questi nomi in codice VBA.

Il codice seguente inserisce il testo "ciao" nella cella A1 del foglio di lavoro il cui nome in codice è Foglio1

```
Sheet1.Range ("A1"). Value = "ciao"
```

Il codice seguente inserisce il testo "ciao" nella cella A1 del foglio di lavoro il cui nome foglio è Foglio1

```
Worksheets ("Foglio1"). Range ("A1"). Value = "ciao"
```

Il codice seguente inserisce il testo "ciao" nella cella A1 del primo foglio di lavoro

```
Worksheets (1). Range ("A1"). Value = "ciao"
```

Attivare o selezionare un foglio di lavoro

Il foglio di lavoro attivo è il foglio di lavoro che si sta attualmente visualizzando o lavorare e la proprietà *Workbook.ActiveSheet* restituisce il foglio attualmente attivo in una cartella di lavoro: Sintassi: *WorkbookObject.ActiveSheet*. Per rendere un foglio attivo si deve utilizzare il metodo *Worksheet.Activate* con questa sintassi: ***WorksheetObject.Activate***. Si noti che è possibile selezionare più fogli di lavoro, ma si può attivare solo un singolo foglio. Per selezionare un foglio di lavoro si utilizza il metodo *Worksheet.Select* con questa sintassi: *WorksheetObject.Select (Replace)*. *Replace* è un argomento facoltativo, usato solo quando si utilizza il metodo Select per i fogli (il metodo Select è utilizzato anche per selezionare le celle) e ponendolo a True, questo argomento sostituirà la selezione precedente, mentre con False includerà la selezione precedente (estendendo così la selezione corrente per includere i fogli selezionati in precedenza) mentre omettendo la specifica, l'argomento sarà posto di default a True. Si utilizza il metodo *Worksheets.Select* per selezionare tutti i fogli in una cartella di lavoro.
Sintassi: *WorksheetsObject.Select (Replace)*.

Esempio: Selezionare fogli singoli o multipli considerando una cartella di lavoro contenente tre fogli di lavoro

```
Sub seleziona1()
ActiveWorkbook.Worksheets(2).Select 'Seleziona e attiva il secondo foglio di lavoro Foglio2
MsgBox ActiveWorkbook.ActiveSheet.Name 'Restituisce il foglio di lavoro attivo Foglio2
'Selezionare più fogli di lavoro nella cartella utilizzando un array di nomi di foglio
'dove il primo foglio nella matrice è Foglio3 che diventa il foglio di lavoro attivo
ActiveWorkbook.Sheets(Array("Foglio3", "Foglio1")).Select
MsgBox ActiveWorkbook.ActiveSheet.Name 'Restituisce il foglio di lavoro attivo Foglio3
'Seleziona e attivare il secondo foglio di lavoro Foglio2, la selezione precedente
'viene sostituita quindi questo è l'unico foglio di lavoro selezionato
ActiveWorkbook.Worksheets("Foglio2").Select
MsgBox ActiveWorkbook.ActiveSheet.Name 'Restituisce il foglio di lavoro attivo Foglio2
Dim i As Integer 'Selezionare più fogli di lavoro , tranne l'ultimo
For i = 1 To ThisWorkbook.Worksheets.Count - 1
ActiveWorkbook.Worksheets(i).Select (False)
Next i
'Restituisce il foglio di lavoro attivo Foglio2, che era il foglio di lavoro attivo prima del ciclo For
MsgBox ActiveWorkbook.ActiveSheet.Name
End Sub
```

Esempio: Selezionare e attivare i fogli di lavoro, considerando una cartella di lavoro che contenente 4 fogli, vale a dire Foglio1, Foglio2, Foglio3 e Foglio4

```
Sub seleziona2()
ActiveWorkbook.Worksheets(1).Select 'Seleziona e attiva il primo foglio (Foglio1)
MsgBox ActiveWorkbook.ActiveSheet.Name 'Restituisce il foglio di lavoro attivo (Foglio1)
'Seleziona il secondo foglio (Foglio2), senza attivarlo e senza deselezionare la selezione precedente
ActiveWorkbook.Worksheets(2).Select (False)
MsgBox ActiveWorkbook.ActiveSheet.Name 'Restituisce il foglio di lavoro attivo (Foglio1)
'Seleziona il terzo foglio (Foglio3), senza attivarlo e senza deselezionare la selezione precedente
ActiveWorkbook.Worksheets(3).Select (False)
MsgBox ActiveWorkbook.ActiveSheet.Name 'Restituisce il foglio di lavoro attivo (Foglio3)
'Attiva il terzo foglio di lavoro (Foglio3), senza deselezionare la selezione precedente
 `perché uno dei fogli selezionati viene attivato
ActiveWorkbook.Worksheets(3).Activate
MsgBox ActiveWorkbook.ActiveSheet.Name 'Restituisce il foglio di lavoro attivo (Foglio3)
'I fogli di lavoro (Foglio1, Foglio2 e Foglio3) sono attualmente selezionati
Dim ws As Worksheet
For Each ws In ActiveWindow.SelectedSheets
MsgBox ws.Name
Next
'Attiva e seleziona il quarto foglio di lavoro (Foglio4), che è l'unico foglio selezionato ora,
 `deselezionando la selezione precedente, perché il foglio attivo non è uno dei fogli di lavoro
 `selezionati in precedenza
ActiveWorkbook.Worksheets(4).Activate
MsgBox ActiveWorkbook.ActiveSheet.Name 'Restituisce il foglio di lavoro attivo (Foglio4)
End Sub
```

Esempio: Selezionare tutti i fogli, considerando una cartella di lavoro contenente 3 fogli di lavoro, Foglio1, Foglio2 e Foglio3

```
Sub seleziona3()
ActiveWorkbook.Worksheets.Select 'Seleziona tutti i fogli di lavoro nella cartella di lavoro attiva
MsgBox ActiveWorkbook.ActiveSheet.Name 'Il primo foglio diventa il foglio attivo -Foglio1
'Seleziona tutti i fogli della cartella di lavoro attiva utilizzando una matrice di nomi
ActiveWorkbook.Sheets(Array("Foglio2", "Foglio3", "Foglio1")).Select
'Il primo foglio nella matrice diventa il foglio attivo e restituisce Foglio2
MsgBox ActiveWorkbook.ActiveSheet.Name
'attivare e selezionare il terzo foglio di lavoro (Foglio3), deselezionando tutti gli altri fogli
ActiveWorkbook.Worksheets(3).Activate
MsgBox ActiveWorkbook.ActiveSheet.Name 'Restituisce il foglio di lavoro attivo (Foglio3)
'Modo alternativo per selezionare tutti i fogli della cartella di lavoro attiva
Dim ws As Worksheet
For Each ws In ActiveWorkbook.Worksheets
If ws.Visible Then ws.Select (False)
Next
MsgBox ActiveWorkbook.ActiveSheet.Name 'il foglio attivo rimane (Foglio3)
End Sub
```

La proprietà Window.SelectedSheets

Si utilizzare la proprietà *Window.SelectedSheets* per determinare tutti i fogli selezionati in una finestra specificata con la seguente sintassi:*WindowObject.SelectedSheets*, dove Window (1) si riferisce sempre alla finestra attiva. Una sola finestra (Window Object) è un membro della Collezione di Windows e la raccolta di Windows contiene tutte le finestre dell'Applicazione Excel (raccolta di Windows per l'oggetto Application) o contiene tutte le finestre della cartella di lavoro specificata (raccolta di Windows per oggetto Workbook).

Aggiungere e rinominare Fogli di lavoro

Per creare o aggiungere un nuovo foglio di lavoro in una cartella di lavoro specificato, si utilizzare il metodo *Sheets.Add*, che diventa anche il foglio attivo con questa sintassi: *SheetsObject.Add(Before, After, Count, Type)* . I nomi di default dei fogli sono Foglio1, Foglio2 etc. in ordine da sinistra a destra e tutti gli argomenti sono opzionali. Si utilizza l'argomento *Before* per specificare il foglio prima del quale si desidera aggiungere il nuovo foglio, mentre si utilizza l'argomento *After* per specificare il foglio dopo il quale si desidera aggiungere il nuovo foglio. L'argomento *Count* specifica il numero di fogli che si desidera aggiungere, di default è uno, mentre si utilizza l'argomento *Type* per specificare un valore costante o (costante XlSheetType) che indica il tipo di foglio da aggiungere che possono essere: XlSheetType, xlWorksheet, xlChart, xlExcel4MacroSheet, xlExcel4IntlMacroSheet o xlDialogSheet. Esempi di aggiunta di un foglio

Utilizzando il metodo ***Add*** senza specificare alcun argomento, si aggiunge un nuovo foglio di lavoro prima del foglio attivo e se gli argomenti Before e After vengono omessi il valore predefinito dell'argomento Count è 1

```
ActiveWorkbook.Worksheets.Add
```

Utilizzando il metodo **Add** e specificando l'argomento **After**, si aggiunge un nuovo foglio dopo il foglio di lavoro denominato Foglio2

```
ActiveWorkbook.Worksheets.Add After: = Worksheets ("Foglio2")
```

Utilizzando il metodo Add e specificando i due argomenti After e **Count**, si aggiungono 3 nuovi fogli di lavoro dopo il foglio di lavoro Foglio2

```
ActiveWorkbook.Worksheets.Add After: = Worksheets ("Foglio2"), Count: = 3
```

Utilizzando il metodo Add e specificando i due argomenti After e Count, si aggiungono 2 nuovi fogli di lavoro dopo il foglio di lavoro denominato Foglio2

```
ActiveWorkbook.Worksheets.Add, Worksheets ("Foglio2"), 2
```

Nomi di fogli predefiniti

Abbiamo visto poco sopra come creare una nuova cartella di lavoro utilizzando il metodo *Workbooks.Add* (riferito all'oggetto Workbooks) e a meno che non si specifichi un file Excel come modello per la nuova cartella di lavoro, il metodo Add creerà una nuova cartella di lavoro di Excel con il default di tre fogli bianchi, in cui il numero predefinito di fogli può essere modificato utilizzando la proprietà *Application.SheetsInNewWorkbook*. I tre fogli bianchi avranno di default i nomi Foglio1, Foglio2 e Foglio3, in ordine da sinistra a destra.

Si utilizza la proprietà ***Name*** dell'oggetto Worksheet (Worksheet.Name) per impostare o restituire il nome di un foglio di lavoro Sintassi:*WorksheetObject.Name*. Alcuni esempi di utilizzo della proprietà Nome: Restituisce il nome del primo foglio di lavoro nella cartella di lavoro attiva

```
MsgBox ActiveWorkbook.Worksheets (1). Name
```

Attivare il foglio di lavoro denominato Foglio1 della cartella di lavoro attiva

```
ActiveWorkbook.Worksheets ("Foglio1"). Activate
```

Modificare il nome del foglio di lavoro denominato Foglio1 nella cartella di lavoro attiva prova1

```
ActiveWorkbook.Worksheets ("Foglio1"). Name = "prova1"
```

Utilizzando il metodo Add e specificando l'argomento Before si aggiunge un nuovo foglio di lavoro prima del foglio di lavoro Foglio2, e utilizzando la proprietà Name, si rinomina il nuovo foglio di lavoro pippo

```
ActiveWorkbook.Worksheets.Add (Before: = Worksheets ("Foglio2")). Name = "pippo"
```

Restituire i nomi di tutti i fogli (fogli di lavoro, grafici, fogli macro e fogli di dialogo) in ThisWorkbook

```
Dim i As Integer
For i = 1 To ThisWorkbook.Sheets.count
MsgBox ThisWorkbook.Sheets (i). Name
Next i
```

Restituire i nomi di tutti i fogli di lavoro (esclusi grafici, macro e fogli di dialogo) nella cartella di lavoro attiva

```
Dim ws As Worksheet
For Each ws In ActiveWorkbook.Worksheets
MsgBox ws.Name
Next
```

Esempio: Aggiungere un nuovo foglio, e consultarlo utilizzando la sua variabile oggetto. Per utilizzare la variabile oggetto attraverso la procedura, verrà dichiarata pubblica (Public), nella prima linea del modulo.

```
Sub foglio1()
Dim wsNew As Worksheet
Set wsNew = Worksheets.Add
wsNew.Name = "Pippo1"
'Riferimento al nuovo foglio - il nuovo foglio diventa il foglio attivo
Worksheets.Add
ActiveSheet.Name = "Pippo2"
End Sub
```

Copiare un foglio di lavoro

Si utilizza il metodo ***Worksheet.Copy*** per copiare un foglio di lavoro e posizionarlo, prima o dopo un altro foglio in una cartella di lavoro, o per creare una nuova cartella di lavoro contenente il foglio copiato utilizzando il metodo *Sheets.Copy* per copiare un foglio usando la seguente sintassi: *WorksheetObject.Copy (Before, After)*. Entrambi gli argomenti *Before* e *After* sono opzionali, ed è possibile specificare solo uno alla volta di questi e questi argomenti fanno riferimento al foglio *prima* o *dopo* dove il foglio copiato sarà posto, e omettendo entrambi gli argomenti si creerà una nuova cartella di lavoro contenente il foglio copiato. È possibile copiare un foglio di lavoro in una posizione all'interno della stessa cartella di lavoro o in un'altra cartella di lavoro.

Copiare il foglio denominato Foglio1 e posizionarlo nella cartella attiva prima di Foglio3

```
Worksheets ("Foglio1"). Copy Before: = Sheets ("Foglio3")
```

Copiare il foglio denominato Foglio1 e posizionarlo nella cartella attiva dopo il Foglio3

```
Worksheets ("Foglio1"). Copy After: = Sheets ("Foglio3")
```

Copiare Foglio1 dalla cartella di lavoro attiva e metterlo prima del foglio pippo nella cartella prova.xlsm

```
Worksheets ("Foglio1"). Copy Before: = Workbooks ("prova.xlsm"). Sheets ("pippo")
```

Copiare Foglio1 dalla cartella attiva in una nuova cartella di lavoro

```
Worksheets ("Foglio1"). Copy
```

Spostare o Cambiare Sequenza al foglio

Si utilizza il metodo ***Worksheet.Move*** per spostare un foglio di lavoro e posizionarlo, prima o dopo un altro foglio in una cartella di lavoro, o per creare una nuova cartella di lavoro contenente il foglio spostato (si utilizza il metodo *Sheets.Move* per spostare un foglio) con questa sintassi: *WorksheetObject.Move (Before, After)*. Entrambi gli argomenti *Before* e *After* sono opzionali, ed è possibile specificare solo uno di questi alla volta, inoltre questi argomenti fanno riferimento al foglio *prima* o *dopo* dove verrà inserito il foglio spostato, e omettendo entrambi gli argomenti si creerà una nuova cartella di lavoro contenente il foglio spostato. È possibile spostare un foglio di lavoro in una posizione all'interno della stessa cartella di lavoro o ad un'altra cartella di lavoro e lo spostamento di un foglio di lavoro modifica la sequenza dei fogli di lavoro, nelle schede che appaiono in una cartella di lavoro, ma non in Esplora progetti in VBE.

Spostare il foglio denominato Foglio1 nella cartella attiva e posizionarlo prima del Foglio3

```
Worksheets ("Foglio1"). Move Before: = Sheets ("Foglio3")
```

Spostare il foglio denominato "Foglio1" nella cartella attiva e posizionarlo dopo il "Foglio3"

```
Worksheets ("Foglio1"). Move After: = Sheets ("Foglio3")
```

Spostare il Foglio1 dalla cartella di lavoro attiva e metterlo prima del foglio pippo nella cartella prova.xlsm

```
Worksheets ("Foglio1"). Move Before: = Workbooks ("prova.xlsm"). Sheets ("pippo")
```

Spostare il Foglio1 dalla cartella attiva, e creare una nuova cartella di lavoro che contiene il foglio spostato

```
Worksheets ("Foglio1"). Move
```

Nascondere o rendere visibile un foglio di lavoro

Per nascondere un foglio di lavoro o per renderlo visibile si utilizzare la proprietà *Worksheet.Visible* (*Sheets.Visible*), utilizzando semplicemente il codice*Worksheets ("Foglio1"). Visible = False* per nascondere e *Worksheets ("Foglio1"). Visible = True* per renderlo visibile. L' enumerazione

- ***XlSheetVisibility*** viene utilizzata per impostare o restituire un valore che indica se il foglio è visibile o nascosto
- ***xlSheetHidden*** si nasconde un foglio di lavoro
- ***xlSheetVeryHidden*** si nasconde un foglio di lavoro che può essere reso visibile solo attraverso codice VBA
- ***xlSheetVisible*** si renderà visibile il foglio.

Esempio: Utilizzare i metodi per nascondere, rendere visibile o rendere molto nascosto il foglio1

```
Sub nascondi1()
Worksheets("Foglio1").Visible = False 'Nasconde Foglio1
'Restituisce il valore 0, che indica che Foglio1 è nascosto
MsgBox Worksheets("Foglio1").Visible
Worksheets("Foglio1").Visible = True 'Rende visibile Foglio1
'Restituisce il valore -1, che indica che Foglio1 è visibile
MsgBox Worksheets("Foglio1").Visible
Worksheets("Foglio1").Visible = 2 'Rende Foglio1 molto nascosto
'Restituisce il valore 2, Foglio1 è molto nascosto
MsgBox Worksheets("Foglio1").Visible
Worksheets("Foglio1").Visible = xlSheetVisible 'Rende visibile Foglio1
'Restituisce il valore -1, che indica che Foglio1 è visibile
MsgBox Worksheets("Foglio1").Visible
End Sub
```

Esempio: Nascondere tutti i fogli tranne l'ultimo, tenendo presente che tutti i fogli in una cartella di lavoro non possono essere nascosti

```
Sub nascondi2()
    Dim oSh As Object, i As Long
Sheets(Sheets.Count).Visible = True
For i = 1 To Sheets.Count - 1
Sheets(i).Visible = xlSheetHidden
Next i
For Each oSh In Sheets
oSh.Visible = xlSheetVisible
Next
End Sub
```

Rimuovere o eliminare fogli

Si utilizzare il metodo ***Worksheet.Delete*** per eliminare o rimuovere un foglio di lavoro in una cartella di lavoro (*Sheets.Delete*) con la seguente sintassi:*WorksheetObject.Delete* e usando questo metodo verrà visualizzata una finestra di dialogo che chiede all'utente di confermare o annullare l'eliminazione. Con l'impostazione della proprietà *Application.DisplayAlerts* a False non verrà visualizzata nessuna richiesta o avviso e in questo caso una risposta predefinita sarà scelta da Excel. Questa proprietà viene ripristinata al valore predefinito (True) dopo che la procedura è termina. Utilizzare il codice*Application.DisplayAlerts = False* per eliminare un foglio di lavoro senza visualizzare nessun avviso

Cancellare il Foglio1 nella cartella di lavoro attiva, senza visualizzare nessun avviso

```
Application.DisplayAlerts = False
Sheets ("Foglio1"). Delete
```

Fogli di lavoro e Layout di pagina

Vediamo ora brevemente alcune opzioni per il layout di pagina di un foglio di lavoro, e come personalizzare le viste della cartelle di lavoro. Per le impostazioni di pagina di un foglio di lavoro si utilizza la proprietà *Worksheet.PageSetup* che si occupa degli attributi della pagina per il foglio di lavoro come l'orientamento, i margini, le dimensioni della carta, e così via. Usando la proprietà PageSetup restituisce un oggetto *PageSetup* e gli attributi sono impostati come proprietà dell'oggetto cioè Orientaton, PrintArea, LeftMargin, RightMargin, etc che sono le proprietà dell'oggetto PageSetup. Vedere il seguente esempio sull'utilizzo di queste proprietà per impostare gli attributi della pagina.

```
Sub setup_pagina()
'impostare gli attributi e poi stampare il foglio
With Worksheets("Foglio3")
 'impostare le proprietà dell'oggetto PageSetup
With .PageSetup
.PrintTitleRows = Rows(1).Address
.Orientation = xlLandscape
.Zoom = 90
.PrintArea = "$D$1:$R$50"
'inserire il numero di pagina nell'intestazione e alllineato al centro
.CenterHeader = "&P"
'Imposta il primo numero di pagina da utilizzare durante la stampa del foglio di lavoro
.FirstPageNumber = 2
.PaperSize = xlPaperA4
.LeftMargin = Application.InchesToPoints(0.25)
.RightMargin = Application.InchesToPoints(0.5)
.TopMargin = Application.InchesToPoints(1)
.BottomMargin = Application.InchesToPoints(0.75)
.HeaderMargin = Application.InchesToPoints(0.5)
.FooterMargin = Application.InchesToPoints(0.25)
'Visualizza la griglia delle celle quando il foglio viene stampato
.PrintGridlines = True
End With
 'stampa il foglio
.PrintOut
End With
End Sub
```

E' possibile visualizzare un'anteprima di come il foglio di lavoro verrà stampato utilizzando il metodo *Worksheet.PrintPreview* che presenta questa sintassi:*WorksheetObject.PrintPreview (EnableChanges)*. E' facoltativo specificare l'argomento *EnableChanges*, che accetta un valore booleano (il valore predefinito è True), per consentire o non consentire all'utente di modificare le opzioni di impostazione della pagina (es. orientamento della pagina, il ridimensionamento, i margini, etc) disponibili in anteprima di stampa. È inoltre possibile applicare il metodo *PrintPreview* a un oggetto cartella di lavoro o a un oggetto Range, per visualizzare un'anteprima di come verrà eseguita la stampa. Esempio – PrintPreview

```
Sub anteprima()
'Visualizzare un'anteprima di stampa del foglio attivo della cartella di lavoro "prova.xlsm,
 `non consentendo all'utente di cambiare le opzioni di pagina disponibili in anteprima di stampa
Workbooks("prova.xlsm").PrintPreview EnableChanges:=False
'Visualizzare un'anteprima di stampa del Foglio3 della cartella "prova.xlsm, permettendo
 `all'utente di cambiare le opzioni di pagina disponibili in anteprima di stampa.
Workbooks("prova.xlsm").Worksheets("Foglio3").PrintPreview EnableChanges:=True
End Sub
```

Inoltre utilizzando la proprietà *Worksheet.DisplayPageBreaks* è possibile visualizzare sia le interruzioni di pagina automatiche che manuali su un foglio di lavoro.Questa è un'impostazione booleana che può essere impostata solo se è installata una stampante e con il codice *ActiveSheet.DisplayPageBreaks = True* verranno visualizzate le interruzioni di pagina sul foglio attivo. Si utilizza la proprietà *Window.View*, per impostare o restituire una cartella di lavoro (foglio di lavoro attivo) come verrà mostrata nella finestra, utilizza questa sintassi: *WindowObject.View*. Si possono avere tre impostazioni per questa struttura:

- ***xlNormalView*** *(valore 1)*
- ***xlPageBreakPreview*** *(valore 2)*
- ***xlPageLayoutView*** *(valore 3)*

Che rispettivamente sono le visualizzazioni della cartella di lavoro definite 'Normale', 'Layout di pagina' e 'interruzione di pagina'. Vedi sotto esempio

```
Sub vista1()
Worksheets("Foglio3").Activate 'Attivare il foglio di lavoro
With ActiveSheet
.Rows(9).PageBreak = xlPageBreakManual `si imposta la posizione di un'interruzione manuale
.Columns("G").PageBreak = xlPageBreakManual
With .PageSetup
.Orientation = xlPortrait
.Zoom = 100
.PrintArea = "$A$1:$L$17"
.CenterHeader = "&P" 'Numero di pagina da stampare allineata al centro nell'intestazione
.FirstPageNumber = 1
End With
End With
ActiveWindow.View = xlPageBreakPreview
ActiveSheet.Rows(9).PageBreak = xlPageBreakNone 'Reset alla visualizzazione normale
ActiveSheet.Columns("G").PageBreak = xlPageBreakNone
ActiveSheet.DisplayPageBreaks = False
ActiveSheet.PageSetup.PrintArea = "" 'l'Intero foglio diventa l'area di stampa
ActiveWindow.View = xlNormalView  'Impostare la visualizzazione normale
ActiveWindow.View = xlPageLayoutView 'Impostare la visualizzazione Layout di pagina
End Sub
```

Nascondere la barra della formula

```
Application.DisplayFormulaBar = False
```

Impostare Excel in modalità a schermo intero

```
Application.DisplayFullScreen = True
```

Nascondere le intestazioni del foglio di lavoro

```
ActiveWindow.DisplayHeadings = False
```

Nascondere la visualizzazione della griglia in un foglio di lavoro

```
ActiveWindow.DisplayGridlines = False
```

Blocca riquadri in un foglio di lavoro specificato

```
Range ("C2"). Select
ActiveWindow.FreezePanes = True
```

Spesso è possibile utilizzare i seguenti codici per togliere la barra della formula da tutti i fogli, o all'attivazione del foglio di lavoro, oppure in altri contesti sfruttando svariati eventi e metodi come si vede dagli esempi sotto riportati

```
Private Sub Workbook_Activate ()
Application.DisplayFormulaBar = False
End Sub

Private Sub Workbook_Deactivate ()
Application.DisplayFormulaBar = True
End Sub

Private Sub Workbook_Open ()
Dim ws As Worksheet, wn As Window
For Each ws In ActiveWorkbook.Worksheets
ws.activate
For Each wn In ActiveWorkbook.Windows
wn.DisplayHeadings = False
Next
Next
Sheets("Foglio1").Activate
End Sub
```

Calcolo nei Fogli di lavoro

La proprietà *Application.Calculation* restituisce o imposta la modalità di calcolo di Excel e dispone di 3 impostazioni:

- *xlCalculationAutomatic* - (Default) ricalcolo automatico come i dati vengono immessi nelle celle
- *xlCalculationSemiautomatic* - Ricalcolo automatico ad eccezione delle tabelle di dati
- *xlCalculationManual* - Il calcolo viene fatto solo quando richiesto dall'utente facendo clic su "Calcola ora" o "Calcola foglio" o premendo F9, oppure con codice VBA.

Impostare la modalità di calcolo manuale:

```
Application.Calculation = xlCalculationManual
```

Si applicare il metodo *Calculate* a un oggetto foglio per calcolare un foglio di lavoro specifico in una cartella di lavoro, oppure si può applicare questo metodo a un oggetto *Application* per calcolare tutte le cartelle di lavoro aperte, oppure è possibile applicare questo metodo a un intervallo di celle di un foglio di lavoro.

Metodo Calcola applicabile all'oggetto Application, calcola tutte le cartelle che sono aperte

```
Application.Calculate
Calculate
```

Il metodo Calcola applicabile all'oggetto foglio di lavoro; calcolare un foglio di lavoro denominato Foglio1

```
Application.Worksheets ("Foglio1"). Calculate
Worksheets ("Foglio1"). Calculate
```

Il metodo Calcola applicabile all'intervallo di celle specificato in un foglio di lavoro

```
Worksheets ("Foglio1"). Range ("A5: B6"). Calculate
```

Calcolare l'intera colonna (colonna A) in un foglio di lavoro

```
Worksheets ("Foglio1"). Columns (1). Calculate
```

Calcola celle non contigue nel foglio di lavoro attivo

```
Range ("A5, A6, B7, B20"). Calculate
```

In Excel, la modalità di calcolo predefinita è la Modalità Automatica (*Application.Calculation = xlCalculationAutomatic*), in cui Excel calcola automaticamente ogni cella come si entra nel foglio, quando Excel è in modalità manuale (*Application.Calculation = xlCalculationManual*), il calcolo viene effettuato solo quando richiesto dall'utente, facendo clic su "Calcola Ora" o premendo F9 o cambiando la modalità di calcolo. In modalità automatica, per ciascun nuovo valore immesso da una macro, Excel ricalcolerà tutte le celle interessate dal nuovo valore rallentando notevolmente l'esecuzione del codice VBA, soprattutto nel caso di grandi macro i cui calcoli sono significativi.

Per accelerare una macro e rendere l'esecuzione più veloce ed efficiente, è tipico disattivare il calcolo automatico all'inizio della macro e ricalcolare il foglio di lavoro specifico utilizzando il metodo Calculate all'interno della macro. Il seguente esempio disattiva gli aggiornamenti dello schermo e i calcoli automatici e utilizza il metodo Calculate, durante l'esecuzione del codice VBA

```
Sub Calcola()
'Disattivare Aggiornamenti schermo e calcoli automatici
Application.ScreenUpdating = False
Application.Calculation = xlCalculationManual
'Inserisci qui il tuo codice
'Utilizzare il metodo Calculate per calcolare tutte le cartelle di lavoro aperte
Application.Calculate
'Attivare gli aggiornamenti dello schermo e calcoli automatici
Application.ScreenUpdating = True
Application.Calculation = xlCalculationAutomatic
End Sub
```

A volte si può impostare la proprietà *Application.CalculateBeforeSave* a True per calcolare le cartelle di lavoro prima di salvarle su disco se la struttura di calcolo è stata impostata su manuale specificando *xlCalculationManual*. La proprietà CalculateBeforeSave accetta valori booleani (Ture o False) e non è influenzata dalla modifica della proprietà di calcolo. In questo caso

```
Application.Calculation = xlCalculationManual
Application.CalculateBeforeSave = True
```

Si imposta la proprietà *Worksheet.EnableCalculation* a True, per ricalcolare automaticamente il foglio di lavoro quando questa proprietà è impostata su False, Excel non ricalcola il foglio, e quando l'impostazione viene modificata da False a True, Excel esegue il ricalcolo. Si noti che questa proprietà non viene mantenuta quando la cartella di lavoro viene chiusa, e tornerà al suo valore predefinito (True) per l'apertura della cartella di lavoro e, quindi, sarà necessario resettare con il codice VBA ogni volta che si apre la cartella di lavoro. Per controllare i calcoli per ogni foglio di lavoro, è possibile impostare la proprietà *EnableCalculation* a False per i fogli di lavoro che non si desidera calcolare, e quindi impostare la proprietà *Application.Calculation* a*xlCalculationAutomatic*. Si noti che l'esecuzione del metodo Calculate su un foglio di lavoro per il quale la proprietà EnableCalculation è impostata su False, non calcolerà tale foglio di lavoro. Il seguente codice calcola automaticamente tutti i fogli tranne Foglio1

```
Worksheets ("Foglio1"). EnableCalculation = False
Application.Calculation = xlCalculationAutomatic
```