Metodi e Proprietà della Cartella di Lavoro

L'oggetto **WorkBook** è il "*figlio*" dell'oggetto Application nella gerarchia degli oggetti di VBA, e rappresenta una singola cartella di lavoro di Excel all'interno dell'applicazione e si riferisce ad un insieme di tutte le cartelle di lavoro attualmente aperte in Excel. Si noti che mentre si lavora con cartelle di lavoro, si utilizzeranno Proprietà e Metodi dell'oggetto cartella di lavoro e per fare riferimento o restituire un oggetto Workbook (singola cartella di lavoro), possono essere utilizzate le seguenti proprietà.

Proprietà Workbooks.Item

L' **Item** dell'oggetto Workbooks si riferisce ad una singola cartella di lavoro in una collezione e viene rappresentata con questa sintassi: *WorkbooksObject.Item (Index)*, dove *Index* è il nome della cartella di lavoro o il numero di indice che sarebbe anche possibile omettere utilizzando una sintassi come:*WorkbooksObject (WorkbookName)* o *WorkbooksObject (IndexNumber)*. Il numero di indice inizia da 1 per il primo foglio aperto o creato e si incrementa per ogni successiva cartella di lavoro, incluse quelle nascoste. L'ultima cartella di lavoro viene restituita dalla proprietà Count in questo modo: *Workbooks.Count*. La Proprietà *WorkbooksObject.Count*, restituisce il numero di cartelle di lavoro. Consultare gli esempi sotto riportati di utilizzo di questa proprietà.

Esempio: Chiudere la cartella di lavoro denominata "VBA_1.xlsm"

```
Workbooks.Item ("VBA_1.xlsm"). Close
'Oppure
Workbooks ("VBA_1.xlsm"). Close
```

Esempio: Chiudere la cartella di lavoro aperta dopo averla salvata

```
Workbooks(Workbooks.Count).Close  SaveChanges:=True
```

Esempio: Restituisce il nome della cartella di lavoro con indice 2

```
MsgBox Workbooks(2).Name
```

Esempio: Restituisce il nome dell'ultima cartella di lavoro aperta o creata

```
MsgBox Workbooks(Workbooks.Count).Name
```

Esempio: Restituire i nomi di tutte le cartelle di lavoro aperte:

```
Sub workbookNames()
Dim i As Integer
For i = 1 To Workbooks.Count
msgbox Workbooks(i).Name
Next i
End Sub
```

Esempio: Impostare una variabile in una cartella di lavoro:

```
Sub workbookVariable()
Dim wb As Workbook, i As Integer
Set wb = Workbooks("Excel_1.xlsx")
For i = 1 To wb.Worksheets.Count
MsgBox wb.Worksheets(i).Name
Next i
End Sub
```

Proprietà ActiveWorkbook

Questa proprietà restituisce la cartella di lavoro attiva, cioè la cartella di lavoro nella finestra attiva con questa sintassi: *ApplicationObject.ActiveWorkbook*, esempio:

```
MsgBox "il nome della cartella attiva è" & ActiveWorkbook.Name
```

Proprietà ThisWorkbook

Questa proprietà viene utilizzata solo dall'interno dell'applicazione Excel e restituisce la cartella di lavoro in cui il codice viene eseguito al momento. Sintassi: *ApplicationObject.ThisWorkbook*, esempio:

```
MsgBox "Il nome di questa cartella di lavoro è" & ThisWorkbook.Name
```

Si noti che sebbene il più delle volte ActiveWorkbook è la stessa ThisWorkbook, ma potrebbe non essere sempre così, la cartella di lavoro attiva può essere diversa da quella in cui viene eseguito il codice, come illustrato dal seguente esempio di codice.

Esempio: Aprire due file della cartella di lavoro di Excel ("prova1.xlsm" e "prova2.xlsm") in un'unica istanza (questo consentirà a tutte le cartelle di lavoro di accedere alla macro), Inserire il codice nella cartella di lavoro "prova1.xlsm", che è anche la cartella di lavoro attiva

```
Sub ActiveWorkbook_ThisWorkbook()
MsgBox "Il nome della cartella attiva è " & ActiveWorkbook.Name
'Restituisce "prova1.xlsm"
MsgBox " Il nome di questa cartella di lavoro è " & ThisWorkbook.Name
'attiva "prova2.xlsm"
Workbooks("prova2.xlsm").Activate
'ritorna a "prova2.xlsm", mentre ThisWorkbook rimane su "prova1.xlsm"
MsgBox "Il nome della cartella attiva è " & ActiveWorkbook.Name
'ritorna a "prova1.xlsm"
MsgBox " Il nome di questa cartella di lavoro è " & ThisWorkbook.Name
'attiva "prova1.xlsm"
Workbooks("prova1.xlsm").Activate
'ritorna a "prova1.xlsm"
MsgBox "Active Workbook's name is " & ActiveWorkbook.Name
MsgBox "Il nome di questa cartella di lavoro è " & ThisWorkbook.Name
End Sub
```

Il Metodo Workbooks.Open

Si utilizza il metodo *Workbooks.Open* per aprire una cartella di lavoro con questa sintassi: *WorkbooksObject.Open(FileName, UpdateLinks, ReadOnly, Format, Password, WriteResPassword, IgnoreReadOnlyRecommended, Origin, Delimiter, Editable, Notify, Converter, AddToMru, Local, CorruptLoad)* L'argomento *FileName* è necessario mentre tutti gli altri argomenti sono facoltativi, vediamo solo alcuni di loro. L'argomento FileName è un valore di tipo String che specifica il nome del file compresa l'estensione e il suo percorso, mentre l'argomento *UpdateLinks* se impostato non aggiornerà link o riferimenti esterni quando la cartella di lavoro è aperta, mentre specificando il valore 3 aggiornerà i link o riferimenti esterni.

L'argomento *ReadOnly* impostato a True apre la cartella di lavoro in modalità di sola lettura e l'argomento *password* è un valore stringa che specifica la password necessaria per aprire una cartella di lavoro protetta da password, omettendo tale argomento verrà richiesto all'utente una password.

L'argomento *WriteResPassword* è un valore stringa che specifica la password necessaria per aprire una cartella di lavoro in scrittura, (es. apertura di questo file senza la password renderà sola lettura), e omettendo tale argomento verrà richiesto all'utente una password. Si noti che quando una cartella di lavoro è aperta a livello di programmazione, le macro sono abilitate di default.

Metodo Workbooks.Add

Si utilizza il metodo Workbooks.Add per creare una nuova cartella di lavoro, che diventa anche la cartella di lavoro usando questa sintassi:*WorkbooksObject.Add (modello)*. Usando questo metodo VBA restituisce un oggetto cartella di lavoro. E 'opzionale specificare il *modello*, in quanto questo argomento è un valore stringa che specifica il nome (e il percorso) di un file di Excel esistente, e in questo caso il file specificato funge da modello per la nuova cartella di lavoro che viene creato, il che significa che la nuova cartella avrà lo stesso contenuto, formattazione, macro e la personalizzazione del file esistente che è specificato. È inoltre possibile definire una costante

- ***xlWBATemplate Enumeration***: Per questo argomento, e in questo caso la cartella di lavoro appena creata conterrà un singolo foglio del tipo che è specificato
- ***xlWBATChart***: Creerà un foglio grafico
- ***xlWBATExcel4MacroSheet***: Crea una versione di Excel 4 con macro
- ***xlWBATExcel4IntlMacroSheet***: Crea una versione di Excel 4 con foglio macro internazionale
- ***xlWBATWorksheet***: Crea un foglio di lavoro.

Tralasciando l'argomento modello si creerà una nuova cartella di lavoro di Excel di default con tre fogli bianchi, in cui il numero predefinito dei fogli può essere modificato/impostato utilizzando la proprietà *Application.SheetsInNewWorkbook*

Il nome predefinito di una nuova cartella di lavoro quando viene creata utilizzando il metodo Add, e l'argomento modello viene omesso, è denominato*CartelN*, dove la prima cartella di lavoro sarà Cartel1, seguita da Cartel2, e così via, quando l'argomento modello è la costante *xlWBATWorksheet*, le cartelle di lavoro sono denominate SheetN e la prima cartella di lavoro creata sarà Foglio1, seguita da Foglio2, e così via

Esempio: Utilizzare il metodo Add per creare una nuova cartella di lavoro

```
Sub WorkbooksAdd()
Dim i As Integer, n As Integer
i = InputBox("Inserire il numero di cartelle da creare")
For n = 1 To i
Workbooks.Add
Next n
End Sub
```

Il Metodo Close

Si utilizza il metodo Close dell'oggetto Workbooks per chiudere tutte le cartelle di lavoro aperte con questa sintassi: *WorkbooksObject.Close* è inoltre possibile impostare la proprietà *DisplayAlerts* su False per non visualizzare alcuna richiesta o avviso alla chiusura di una cartella. Per chiudere tutte le cartelle di lavoro aperte, utilizzare la riga di codice *Workbooks.Close*, si può utilizzare il metodo Close dell'oggetto Workbook per chiudere una singola cartella di lavoro con questa sintassi: *WorkbookObject.Close (SaveChanges, filename, RouteWorkbook)*.

Tutti gli argomenti sono opzionali da specificare e tralasciando l'argomento SaveChanges verrà richiesto all'utente se salvare le modifiche, nel caso in cui siano state fatte, alla cartella di lavoro dopo l'ultimo salvataggio. Impostando l'argomento SaveChanges a True si salveranno tutte le modifiche apportate alla cartella di lavoro, e se impostato su False, la cartella di lavoro si chiude senza salvare le modifiche apportate e in entrambe le impostazioni la cartella di lavoro si chiude senza visualizzare alcuna richiesta per salvare le modifiche.

L'argomento *filename* dovrebbe essere utilizzato per specificare un nome di file per chiudere una cartella di lavoro che ancora non ha un nome di file associato, cioè una nuova cartella di lavoro, altrimenti con l'omissione di questo argomento si richiederà all'utente di specificare il nome del file prima di chiuderlo.
L'argomento nome del file può essere utilizzato come opzione se si desidera salvare una cartella di lavoro esistente, cioè che ha già un nome associato, con un nuovo nome di file.
L'argomento *RouteWorkbook* impostato a True consente di instradare o inviare al destinatario, se è presente una lista di distribuzione. Esempi di utilizzo di questo metodo:

Per chiudere la cartella di lavoro attiva dopo aver salvato le modifiche

```
ActiveWorkbook.Close SaveChanges: = True
'Per chiudere la cartella di lavoro "prova.xlsx", dopo aver salvato le modifiche
Workbooks ("prova.xlsx"). Close SaveChanges: = True
```

Esempio: Vari metodi di apertura e chiusura della cartella di lavoro con salvataggio

```
Sub WorkbookAdd()
Application.DisplayAlerts = False
ChDir ThisWorkbook.Path 'spostarsi dalla directory corrente a ThisWorkbook
Workbooks.Add "prova1.xlsx"
MsgBox " Il nome della Nuova cartella di lavoro e il numero di fogli è: " & ActiveWorkbook.Name & "; "
& ActiveWorkbook.Sheets.count
'salva come file xls di Excel 97, formato 2003
ActiveWorkbook.SaveAs fileName:=ActiveWorkbook.Name, FileFormat:=xlExcel8
ActiveWorkbook.ActiveSheet.Range("A1") = "xlsFile" 'Inserire in A1 del foglio attivo "xlsFile
ActiveWorkbook.Close SaveChanges:=True 'chiude la cartella dopo aver salvato le modifiche
Workbooks.Add 'Creare una nuova cartella di lavoro con tre fogli di lavoro
MsgBox " Il nome della Nuova cartella di lavoro e il numero di fogli è: " & ActiveWorkbook.Name & "; "
& ActiveWorkbook.Sheets.count
'salvare con estensione xlsx (2007)
ActiveWorkbook.SaveAs fileName:=ActiveWorkbook.Name, FileFormat:=xlOpenXMLWorkbook
ActiveWorkbook.ActiveSheet.Range("A1") = "xlsxFile" 'Inserire nella cella A1 del foglio "xlsxFile"
ActiveWorkbook.Close SaveChanges:=True 'chiude la cartella di lavoro dopo aver salvato le modifiche
Application.SheetsInNewWorkbook = 5 'Modificare il numero predefinito di fogli di lavoro a 5:
Workbooks.Add
MsgBox "Il nome della Nuova cartella di lavoro e il numero di fogli è: " & ActiveWorkbook.Name & "; "
& ActiveWorkbook.Sheets.count
ActiveWorkbook.SaveAs fileName:="NewWorkbookSaved.xlsm",  'salvare la nuova cartella come xlsm,
FileFormat:=xlOpenXMLWorkbookMacroEnabled
ActiveWorkbook.ActiveSheet.Range("A1") = "xlsmFile" 'Inserire " xlsmFile " in A1 del foglio attivo
ActiveWorkbook.Close SaveChanges:=True 'chiude la cartella di lavoro dopo aver salvato le modifiche
Workbooks.Add (xlWBATWorksheet) 'Creare una nuova cartella di lavoro con un foglio di lavoro
MsgBox " Il nome della Nuova cartella di lavoro e il numero di fogli è: " & ActiveWorkbook.Name & "; "
& ActiveWorkbook.Sheets.count
ActiveWorkbook.SaveAs fileName:=ActiveWorkbook.Name 'salva la cartella, omettendo il formato file
ActiveWorkbook.ActiveSheet.Range("A1") = "DefaultFileFormat" 'Inserisce "DefaultFileFormat" in A1
ActiveWorkbook.Close SaveChanges:=True 'chiude la cartella di lavoro dopo aver salvato le modifiche
Application.DisplayAlerts = True
End Sub
```

Il Metodo Workbook.SaveCopyAs

Si utilizza il metodo Workbook.SaveCopyAs per salvare una copia della cartella di lavoro e questo metodo non modifica la cartella di lavoro aperta con questa sintassi: *WorkbookObject.SaveCopyAs (filename)*.
L'argomento *filename* è facoltativo e viene utilizzato per specificare il nome del file con cui viene memorizzata la copia della cartella di lavoro. Si noti che il metodo Workbook.SaveCopyAs permette di salvare con lo stesso formato di file, cioè la cartella di lavoro e la sua copia devono avere lo stesso formato di file, altrimenti durante l'apertura della copia salvata otterrete un messaggio - "Il file che si sta cercando di aprire è in un formato diverso da quello specificato dall'estensione del file ... " ed è possibile che il file non possa venire aperto.

Esempio: Utilizzare il metodo Workbook.SaveCopyAs per salvare una copia della cartella di lavoro attiva

```
Sub WorkbookSaveCopyAs1()
Dim LastRow As Long
'Determinare l'ultima riga scritta nella colonna "A"
LastRow = ActiveWorkbook.ActiveSheet.Range("A" & Rows.count).End(xlUp).Row
'Salvare una copia della cartella di lavoro se i dati in una colonna del foglio attivo è
maggiore di un numero specifico di righe
If LastRow >= 100 Then
'Salvare una copia della cartella di lavoro attiva specificando un nome di file
ActiveWorkbook.SaveCopyAs "C:\Document\Test\Copia_1.xlsm"
End If
'La cartella di lavoro corrente rimane la cartella attiva, la copia è stata salvata e chiusa
MsgBox ActiveWorkbook.Name
End Sub
```

Esempio: Utilizzare il metodo Workbook.SaveCopyAs per salvare una copia di ThisWorkbook con un nome univoco ogni volta.

```
Sub WorkbookSaveCopyAs2()
Dim fname As String, extn As String, MyStr As String
Dim i As Integer, lastDot As Integer
'cambiare la directory corrente a ThisWorkbook
ChDir ThisWorkbook.Path
'Trovare la posizione del punto per distinguere l'estensione del file
For i = 1 To Len(ThisWorkbook.Name)
If Mid(ThisWorkbook.Name, i, 1) = "." Then
lastDot = i
End If
Next i
'Estensione del file estratta e dot prima estensione
extn = Right(ThisWorkbook.Name, Len(ThisWorkbook.Name) - lastDot + 1)
'nome della cartella estratto escluso l'estensione dal punto
MyStr = Left(ThisWorkbook.Name, lastDot - 1)
'specificare il nome per la copia. Parte del nome del file renderà il nome univoco
fname = MyStr & "_" & Format(Now(), "yyyy-mm-dd hh-mm-ss AMPM") & extn
'salvare una copia di ThisWorkbook specificando un nome di file
ThisWorkbook.SaveCopyAs fname
'la cartella di lavoro corrente rimane la cartella attiva, la copia salvata e chiusa
MsgBox ActiveWorkbook.Name
End Sub
```

Esempio: Utilizzare il metodo Workbook.SaveCopyAs per salvare una copia della cartella di lavoro specificata.

```
Sub WorkbookSaveCopyAs3()
'Aprire una cartella di lavoro specificata nella directory corrente
Workbooks.Open "C:\Documenti\Test\prova_3.xlsx"
'la cartella di lavoro aperta diventa la cartella attiva di lavoro
MsgBox ActiveWorkbook.Name
'salvare una copia della cartella di lavoro aperta, specificando un nome di file
ActiveWorkbook.SaveCopyAs "C:\Documenti\Test\Copia_di_prova3.xlsx"
'la cartella di lavoro aperta rimane la cartella di lavoro attiva, la copia rimane chiusa
MsgBox ActiveWorkbook.Name
'chiudere la cartella di lavoro aperta - notare che né la cartella di lavoro aperta o la
copia salvata saranno aperti dopo questo comando
ActiveWorkbook.Close
End Sub
```

Il metodo Workbook.Save

Si utilizza il metodo Workbook.Save per salvare una cartella di lavoro specificata con questa sintassi: *WorkbookObject.Save*. Esempio: Salvare la cartella di lavoro denominato "prova_1.xlsm":

```
Workbooks ("prova_1.xlsm"). Save
```

Si utilizza l'argomento *SaveChanges* del metodo *Close* dell'oggetto Workbook, per salvare la cartella di lavoro prima della chiusura. Esempio per chiudere la cartella di lavoro dopo averla salvata:

```
Workbooks(Workbooks.Count).Close SaveChanges:=True
```

Si utilizzare il metodo Workbook.SaveAs per salvare le modifiche della cartella di lavoro in un file separato con questa sintassi: *WorkbookObject.SaveAs (FileName, FileFormat, password, WriteResPassword, ReadOnlyRecommended, CreateBackup, AccessMode, ConflictResolution, AddToMru, TextCodepage, TextVisualLayout , locale)*. Tutti gli argomenti sono opzionali, vediamo solo alcuni di loro.
L'argomento *FileName* è un valore di tipo String che specifica il nome del file compresa l'estensione e il suo percorso, omettendo il percorso il file verrà salvato nella directory corrente, mentre *FileFormat* specifica il formato del file da salvare, il formato predefinito è la versione corrente di Excel, mentre per un file esistente l'impostazione predefinita è il formato del file che era specificato. Si ricorda che in Excel 2007-2010 durante l'utilizzo SaveAs, è necessario specificare il parametro FileFormat per salvare un nome di file con estensione Xlsm se la cartella di lavoro in fase di salvataggio non è un file Xlsm, vale a dire:

```
FileFormat: = xlOpenXMLWorkbookMacroEnabled
```

È possibile specificare una password case-sensitive fino a 15 caratteri per l'apertura del file, utilizzando l'argomento *password* specificando una password per aprire un file, se non si inserisce la password si aprirà in sola lettura utilizzando l'argomento *WriteResPassword* per specificare la password.

Esempio: Salvare una cartella di lavoro esistente con un nuovo nome e formato

```
Sub workbookSaveAs1()
'salvare la cartella di lavoro con indice 2 e con un nuovo nome e formato del file.
'ricordate che tutti i file dovrebbero essere aperti in una singola istanza di Excel.
'Ricordate che in Excel 2007-2010 durante l'utilizzo di SaveAs, è necessario specificare il
'parametro FileFormat per salvare un nome di file con estensione xlsm se la cartella di
'lavoro non è un file xlsm
Workbooks(2).SaveAs "Workbook_cambiato.xlsm"
End Sub
```

Esempio: Salvataggio di una nuova cartella di lavoro:

```
Sub workbookSaveAs2()
'aggiungere una nuova cartella di lavoro
Workbooks.Add
'salvare la nuova cartella di lavoro, che diventa la cartella di lavoro attiva, protetta da
password
ActiveWorkbook.SaveAs fileName:="Nuova_cart.xlsx", Password:="abc123"
'salvare la nuova cartella di lavoro come un file xlsm con password di protezione
'ActiveWorkbook.SaveAs fileName:="Nuova_cart.xlsm",
'FileFormat:=xlOpenXMLWorkbookMacroEnabled, Password:="abc123"
'salvare la nuova cartella di lavoro come file xlsm con password di protezione, ma può
'essere aperta in sola lettura senza fornire la password
ActiveWorkbook.SaveAs fileName:="Nuova_cart.xlsm",
FileFormat:=xlOpenXMLWorkbookMacroEnabled, WriteResPassword:="abc123"
End Sub
```

Si utilizza la proprietà Workbook.Saved per determinare se sono state apportate modifiche alla cartella di lavoro dopo l'ultimo salvataggio con questa sintassi: *WorkbookObject .Saved* che restituisce un valore booleano, dove True indica che la cartella di lavoro non è stata modificata dopo l'ultimo salvataggio. L'impostazione di questa proprietà su True prima di chiudere una cartella di lavoro in cui sono state apportate modifiche non chiederà di salvare la cartella di lavoro e non verranno salvate le modifiche.

Esempio: Impostare la proprietà Workbook.Saved a True per chiudere la cartella di lavoro con nessuna richiesta di salvare le modifiche:

```
Sub workbookSaved()
'Aprire una cartella di lavoro
Workbooks.Open "C:\Documents\prova5.xlsx"
'Workbook.Saved restituirà true a indicare che non sono state apportate modifiche alla
cartella di lavoro dopo che è stata salvata l'ultima volta
MsgBox Workbooks("prova5").Saved
'modifichiamo la cartella di lavoro
Workbooks("prova5.xlsx").ActiveSheet.Range("A1") = "Ciao Mondo!"
'Workbook.Saved restituirà False per indicare che le modifiche sono state apportate alla
'cartella di lavoro dopo che è stata salvata l'ultima volta
MsgBox Workbooks("prova5.xlsx").Saved
'impostare la proprietà Workbook.Saved su true
Workbooks("prova5.xlsx").Saved = True
'chiudere la cartella di lavoro con nessuna richiesta di salvare le modifiche e questo NON
salverà le eventuali modifiche. Se la proprietà Saved non è stata impostata su True si
otterrà un prompt per salvare le modifiche
Workbooks("prova5.xlsx").Close
End Sub
```

Metodo Workbook.Activate

Si utilizza il metodo Workbook.Activate per attivare una cartella di lavoro e se la cartella di lavoro dispone di più finestre, il metodo attiva la prima finestra. Sintassi: *WorkbookObject.Activate* . Per attivare la cartella di lavoro denominato "prova5.xlsx", utilizzare il codice

```
Workbooks ("prova5.xlsx"). Activate
```

Metodo Workbook.PrintPreview

Si utilizza il metodo Workbook.PrintPreview per visualizzare un'anteprima di come verrà stampata la cartella con questa sintassi:*WorkbookObject.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, ecc) disponibili in anteprima di stampa.

Esempio: Utilizzo di PrintPreview

```
Sub PrintPreview()
'Anteprima di stampa del foglio attivo della cartella di lavoro, impedendo all'utente di
'cambiare le impostazione della pagina di anteprima di stampa.
Workbooks("prova5.xlsx").PrintPreview EnableChanges:=False
'anteprima di stampa di "Foglio3" di "prova5.xlsx, permettendo all'utente di cambiare le
'impostazioni di pagina disponibili in anteprima di stampa.
Workbooks("prova5.xlsx").Worksheets("Foglio3").PrintPreview EnableChanges:=True
End Sub
```

Metodo Workbook.SendMail

Molte persone utilizzano Outlook come client per la posta elettronica, è possibile automatizzare Outlook, consentendo maggiori funzionalità di inviare e-mail, lavorando con gli oggetti di Outlook utilizzando VBA in Excel. L'automazione è un processo mediante il quale una applicazione comunica o controlla con un'altra applicazione e un'opzione per inviare e-mail da Excel è quella di utilizzare il metodo Workbook.SendMail. Con il metodo Workbook.SendMail , si utilizza il sistema di posta installato per inviare una cartella di lavoro. Sintassi: *WorkbookObject.SendMail(Recipients, Subject, ReturnReceipt))*.

L'argomento *Recipients* è necessario per specificare un singolo destinatario o più destinatari come testo o un array di stringhe di testo, rispettivamente e l'argomento *Subject* è facoltativo e viene utilizzato per specificare l'oggetto della mail e se si omette questo argomento di default verrà usato il nome della cartella di lavoro come oggetto. L'argomento *ReturnReceipt* è facoltativo, se viene usato si specifica il suo valore, che può essere True o False e indica la richiesta di richiedere una ricevuta di ritorno, il valore predefinito è False. Per inserire il nome del destinatario come testo:

```
ActiveWorkbook.SendMail Recipients:="nome_destinatario"
```

Esempio: Inviare una cartella di lavoro, specificando il nome del destinatario o indirizzo email

```
Sub inviaM_1()
Workbooks.Open ("C:\Documents\Test1\prova1.xlsx")
'specificare il nome del destinatario come testo
ActiveWorkbook.SendMail Recipients:="Gino Primo", Subject:="Ciao",
ReturnReceipt:=True
'specifica indirizzo e-mail di destinazione
ActiveWorkbook.SendMail Recipients:="info@gino.com", Subject:="Ciao",
ReturnReceipt:=True
End Sub
```

Esempio: Inviare cartella di lavoro, specificando più destinatari.

```
Sub inviaM_2()
'Specificare più destinatari
ThisWorkbook.SendMail Array("Gino Primo", "Beppe Secondo")
'specificare più indirizzi e-mail come destinazione
ThisWorkbook.SendMail Array(" info@gino.com", " info@beppe.com")
End Sub
```

Esempio: Inviare un foglio Excel con il metodo Sendmail.

```
Sub inviaM_3()
Dim name1 As String, name2 As String
Dim wb As Workbook
'cambiare la directory corrente a ThisWorkbook, utilizzando ChDir
ChDir ThisWorkbook.Path
'crea una nuova cartella di lavoro con un unico foglio da copiare da ThisWorkbook
ThisWorkbook.Worksheets("Foglio1").Copy
'impostare la variabile wb per la nuova cartella di lavoro, che diventa la cartella attiva
Set wb = ActiveWorkbook
'rinominare il foglio nella nuova cartella di lavoro
wb.Sheets("Foglio1").Name = "Nuovo Foglio"
'salvare la nuova cartella di lavoro con un nome di file, nella cartella predefinita
wb.SaveAs fileName:="NuovoF.xlsx"
'assegnare delle variabili per I destinatari
name1 = "Gino Primo"
name2 = "Beppe Secondo"
'Inserire i nomi dei destinatari in una matrice
wb.SendMail Array(name1, name2),
Subject:=ThisWorkbook.Worksheets("Foglio1").Range("A1"), ReturnReceipt:=False
'chiudere la nuova cartella di lavoro
wb.Close
End Sub
```

Esempio: Inviare la cartella di lavoro, specificando il destinatario in una cella del foglio di lavoro, con l'indicazione della data in oggetto.

```
Sub inviaM_4()
Dim strRec As String
'La cella A14 contiene: info@gino.com
strRec = ThisWorkbook.Sheets("Foglio1").Range("A14").Value
'l'oggetto apparirà come "Si prega di controllare 10/08/2014 10:43:06 "
ActiveWorkbook.SendMail Recipients:=strRec, Subject:="Si prega di controllare " &
Format(Now, "dd/mm/yyyy hh:mm:ss AMPM")
End Sub
```

Esempio: Inviare la cartella di lavoro, specificando più destinatari da intervallo di prospetto.

```
Sub inviaM_5()
Dim MyArr As Variant
'la cella A14 contiene info@gino.com, e la cella A15 contiene info@beppe.com
ActiveWorkbook.SendMail
Recipients:=ThisWorkbook.Sheets("Foglio1").Range("A14:A15").Value
MyArr = ThisWorkbook.Sheets("Foglio1").Range("A14:A15")
ActiveWorkbook.SendMail Recipients:=MyArr
'La cella A10 contiene gino Primo la cella A11 contiene beppe secondo, in A14 è stato
inserito info@gino.com, e in A15 info@beppe.com
'Gli intervalli sono stati nominati "nomi_1"=Range("A10:A11"),
"email_1"=Range("A14:A15")
MyArr = ThisWorkbook.Sheets("Sheet1").Range("nomi_1")
MyArr = ThisWorkbook.Sheets("Sheet1").Range("email_1")
ActiveWorkbook.SendMail Recipients:=MyArr
End Sub
```

Proprietà Workbook.ActiveSheet

Questa proprietà restituisce la scheda attualmente attiva in una cartella di lavoro e si usa con questa sintassi: *WorkbookObject.ActiveSheet*. Se appaiono più finestre per una cartella di lavoro, ActiveSheet potrebbe essere diversa per ogni finestra, se non c'è un foglio attivo, questa proprietà restituisce Nothing. Utilizzare il metodo Activate dell'oggetto foglio di lavoro per attivarne una, cioè attivare un foglio. Per esempio

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

Attiverà il foglio denominato "Foglio3" nella cartella di lavoro attiva, il cui nome verrà visualizzato da

```
MsgBox ActiveWorkbook.ActiveSheet.Name
```

Proprietà Workbook.Name

Questa proprietà restituisce il nome di una cartella di lavoro (valore stringa).
Sintassi: *WorkbookObject.Name* ed è di sola lettura e non è possibile modificare il nome della cartella di lavoro. È tuttavia possibile utilizzare il metodo Workbook.SaveAs per salvare una cartella di lavoro esistente (in un nuovo file) con un nuovo nome o salvare una nuova cartella di lavoro con nome.

```
Sub WorkbookName ()
'restituire i nomi di tutte le cartelle di lavoro aperte
 Dim i As Integer, conta As Integer
'restituisce il numero delle cartelle di lavoro
count = Workbooks.Count
'Restituisce i nomi di tutte le cartelle
For i = 1 To count
MsgBox Workbooks(i).Name
Next i
End Sub
```

Proprietà Workbook.ActiveChart

Questa proprietà restituisce il grafico attualmente attivo, che può essere un foglio grafico o un grafico incorporato. Sintassi: *WorkbookObject.ActiveChart*, se non c'è un grafico attivo, questa proprietà restituisce Nothing. Si utilizza il metodo Activate della 'oggetto Chart' o 'oggetto ChartObject' per attivare rispettivamente un foglio grafico o un grafico incorporato, tenendo presente che un foglio grafico è attivo se selezionato dall'utente o attivato utilizzando il metodo Activate. Vedi esempio sotto riportato

Esempio: Illustrare proprietà ActiveChart e metodo Activate.

```
Sub attiva_graf()
ActiveWorkbook.Sheets("Chart1").activate 'Attivare il foglio grafico denominato "Chart1"
'In alternativa, per attivare un foglio grafico: 'ActiveWorkbook.Charts (1). activate
'attivare il grafico incorporato denominato "Grafico 13" in "Foglio2", utilizzando il metodo Activate
ActiveWorkbook.Sheets("Foglio2").ChartObjects("Chart 13").activate
'In alternativa, per attivare il grafico 1 nel "Foglio2" ActiveWorkbook.Sheets ("Foglio2")
ChartObjects (1).Activate, restituisce il grafico attivo - "Foglio2 Chart 13"
MsgBox ActiveWorkbook.ActiveChart.Name
'Aggiungere un titolo al grafico attivo (grafico incorporato denominato "Chart 13" in "Foglio2"):
With ActiveWorkbook.ActiveChart
.HasTitle = True
.ChartTitle.Text = "Ems_Grafico1"
End With
MsgBox ActiveWorkbook.ActiveChart.ChartTitle.Text 'Restituisce il titolo del grafico attivo
End Sub
```

Proprietà Workbook.FileFormat

Questa proprietà restituisce il formato del file della cartella di lavoro. Sintassi: *WorkbookObject.FileFormat* ed è di sola lettura, inoltre è possibile specificare o impostare il formato di file durante il salvataggio di una cartella di lavoro esistente o nuova utilizzando il metodo Workbook.SaveAs (discusso sopra).

Proprietà Workbook.Path

Si utilizza questa proprietà per restituire il percorso (valore stringa) completa della cartella di lavoro Sintassi: *WorkbookObject.Path*.

Proprietà Workbook.Password

Si utilizza questa proprietà per impostare o restituire una password per l'apertura di una cartella di lavoro e si tratta di una proprietà di lettura/scrittura con la seguente sintassi: *WorkbookObject.Password*. Vedere l'esempio sottostante che illustra come impostare una password per aprire una cartella di lavoro e modificare e cancellare una password esistente.

Esempio: Imposta una password per la cartella di lavoro da aprire, cambiare password, eliminare password.

```
Sub pass_cart()
Application.DisplayAlerts = False
Dim fpath As String, fname As String, Pswd As String, newPswd As String
'specificare il nome completo (cioè il percorso e nome) del file da aprire
fpath = "C:\Documents\Test1"
fname = "prova1.xlsx"
'specificare la password per aprire il file
Pswd = "123"
newPswd = "abc"

'aprire la cartella di lavoro senza password
Workbooks.Open fileName:=(fpath & "\" & fname)
'imposta password
ActiveWorkbook.Password = Pswd
'chiudi la cartella salvando le modifiche
ActiveWorkbook.Close SaveChanges:=True

'apri la cartella di lavoro con una password e aggiorna link o riferimenti esterni
Workbooks.Open fileName:=(fpath & "\" & fname), UpdateLinks:=3, Password:=Pswd
'il file verrà salvato senza visualizzare alcuna richiesta
ActiveWorkbook.SaveAs fileName:=(fpath & "\" & fname), Password:=newPswd
'chiudi la cartella salvando le modifiche
ActiveWorkbook.Close SaveChanges:=True

'apri la cartella di lavoro con una password e aggiorna link o riferimenti esterni
Workbooks.Open fileName:=(fpath & "\" & fname), UpdateLinks:=3,
Password:=newPswd
'il file verrà salvato senza visualizzare alcuna richiesta
ActiveWorkbook.SaveAs fileName:=(fpath & "\" & fname), Password:=""
'chiudi la cartella salvando le modifiche
ActiveWorkbook.Close SaveChanges:=True
Application.DisplayAlerts = True
End Sub
```