Find and replace

It is common to use Microsoft Word macros to find and replace text. There are differences between Visual Basic for Applications (VBA) and AppleScript in the find and replace implementation.

Find and replace is global in AppleScript

Unlike the ordinary, global Find and Replace dialog box in the user interface, Word VBA only finds and replaces within a specific part of a document (for example, the main body, a header, or a text box). However, in AppleScript, you can do a global find and replace throughout a document, including headers, footers, and text boxes. Compare the following examples:

VBA

```
Sub ReplaceEverywhere(FindText As String, ReplaceText as String)
Dim oSection As Section
Dim oShape As Shape
Dim oHF As HeaderFooter

           ReplaceInRange FindText, ReplaceText, ActiveDocument.Content
          For Each oShape In ActiveDocument.Shapes
                   If oShape.TextFrame.HasText Then
                            ReplaceInRange FindText, ReplaceText,
                                    oShape.TextFrame.TextRange
                  End If
         Next oShape
          For Each oSection In ActiveDocument.Sections
                   For Each oHF In oSection.Headers
                            ReplaceInRange FindText, ReplaceText,
                                           oHF.Range
                           For Each oShape In oHF.Shapes
                                   If oShape.TextFrame.HasText
                                            Then ReplaceInRange FindText,
                                           ReplaceText, _
                                            oShape.TextFrame.TextRange
                                   End If
                           Next oShape
                  Next oHF
                   For Each oHF In oSection.Footers
                            ReplaceInRange FindText, ReplaceText,
                                   oHF.Range
                           For Each oShape In oHF.Shapes
                                    If oShape.TextFrame.HasText Then
                                            ReplaceInRange FindText,
                                                   ReplaceText,
                                                    oShape.TextFrame.TextRange
                                   End If
                                           Next oShape
                                   Next oHF
                  Next oSection
End Sub

Sub ReplaceInRange(FindText As String, ReplaceText as String, _
                  oRange as Range)
         With oRange.Find
                  .Format = False
                  .Text = FindText
                   .Replacement.Text = ReplaceText
                  .Wrap = wdFindStop
                   .Execute Replace:=wdReplaceAll
         End With
End Sub
```

AppleScript

```
on ReplaceEverywhere(findText, replaceText)
           local theShape, theRange, theHeaderFooters, theHeaderFooter
          tell application "Microsoft Word"
                   --first find and replace in the body (main story) of document
                   set theRange to text object of active document
                    my ReplaceInRange(findText, replaceText, theRange)

                  --nowin all shapes
                   set allShapes to (every shape of active document)
                   repeat with theShape in allShapes
                            set theTextFrame to (text frame of theShape)
                            if has text of (text frame of theShape) then
                                    --note:'text range', not 'text object'
                                           of text frame
                                   set theRange to text range of text frame
                                           of theShape
                                    my ReplaceInRange(findText, replaceText,
                                           theRange)
                          end if
```

```
                    end repeat

                     -- now in the headers and footers of each section
                     set allSections to every section of active document
                     repeat with theSection in allSections
                              set theHeaderFooters to {get header theSection index
                                     header footer primary} &
                                     {get header theSection index
                                     header footer first page} &
                                     {get header theSection index
                                     header footer even pages} &
                                     {get footer theSection index
                                     header footer primary} &
                                     {get footer theSection index
                                     header footer first page} &
                                     {get footer theSection index
                                     header footer even pages}

                              repeat with theHeaderFooter in theHeaderFooters
                                     set theRange to text object
                                             of theHeaderFooter
                                     my ReplaceInRange(findText,
                                             replaceText, theRange)

                                     --now in their shapes
                                     set allShapes to (every shape
                                             of theHeaderFooter)
                                     repeat with theShape in allShapes
                                              if has text of (text frame of theShape)
                                                      then
                                                      --note:'text range', not
                                                              'text object'
                                                      of text frame
                                              set theRange to text range
                                                      of text frame
                                                      of theShape
                                               my ReplaceInRange(findText,
                                                       replaceText, theRange)
                                             end if
                                     end repeat
                             end repeat
                    end repeat
            end tell
    end  ReplaceEverywhere


    on  ReplaceInRange(findText,  replaceText,  theRange)
              tell application "Microsoft Word"
                       set findObject to find object of theRange
                      tell findObject
                               set format to false
                               set content to findText
                                set content of its replacement to replaceText
                               set wrap to find stop
                    end tell
                               execute find findObject replace replace all
            end tell
    end  ReplaceInRange
```

Placing commands and parameters with tell blocks

You can see that in the `ReplaceInRange` handler, which does the work for each **range**, there is a `tell` block to **findObject** (the **findObject** property of the **text range**).
In VBA, the **Execute** method could also be placed inside the `With` block, just like all of the properties, but the equivalent in AppleScript does not seem to work at first with the **execute find** command, even though it takes the same direct parameter (**findObject**) and properties. It works fine when placed outside of the `tell` block and if **findObject** is specifically referenced as the direct parameter.
Commands with direct parameters that are the target of a `tell` block should act on the targeted object without a problem, but they can sometimes be problematic in AppleScript. In fact, you can place **execute find** inside the `tell` block if you include the term `it` to make it clear what the direct object is, as shown in the following example:

```
tell  findObject
          set format to false
          set content to findText
           set content of its replacement to replaceText
          set wrap to find stop
           execute find it replace replace
all
end  tell
```

Classes and properties with the same name

The **text frame** class in the Drawing Suite has a **text range** (not **text object**) property to represent its **range**. Generally speaking, the developers were very careful not to give properties the same name as classes. So various classes have this convention: **findObject** property of class **find**, **font object** property of type **font**, and **text object** property of type **text range**. In this example, the convention was forgotten.
If you ever wanted to run a `whose` filter on the **text range** property of **text frame**, or refer to it in a `tell` block targeted at **text frame**, include the keyword `its`.
There is one more example, shown above, of properties and classes sharing the same name. In the `ReplaceInRange` handler, in the `tell` block, which is directed at **findObject**, `its replacement` is used because **replacement** is also a class name. The script would error here if we did not include `its`, which specifies that the **replacement** property for **findObject** be used and not the application's **replacement** element (class) that would otherwise take precedence.

Copy and paste frequently used handlers

Keep this pair of handlers available, as if they were in a script "library" (a saved script that has no top-level commands, only handlers (subroutines) and perhaps some script properties if needed).

You just copy and paste these two handlers into any script that needs it. When you need a text find and replace, you then call the `ReplaceEverywhere` handler, as in the following example that replaces "hot" with "cold":

```
my  ReplaceEverywhere("hot",  "cold")
```

Traditionally, you put this "top level" command at the top of the script and the handlers at the bottom, but in AppleScript, you can do the opposite. When the script compiles, it "learns" where everything is.

Use my when calling handlers

Strictly speaking, if the call to `ReplaceEverywhere("hot",  "cold")` is not itself inside an application `tell` block, you do not need the `my`. But because you will be calling it from within another Word`tell` block most of the time, you absolutely need the `my`. This means that the script, not the application, is the "parent", so the script will not assume that the term `ReplaceEverywhere` is a keyword of the application, which would fail and error.
The best practice is to always use `my` when calling subroutines.