

GOOGLE HACKS™

**2nd
Edition**
Covers Gmail

Tips & Tools for Smarter Searching



Tara Calishain & Rael Dornfest

*With a new foreword by
Craig Silverstein, Director of Technology, Google*

O'REILLY®

HACK
#63

Google from Word

Add a little Google to Microsoft Word.

You probably use Google a few dozen times a day. If you work a lot within Microsoft Word, using Google usually means switching over to your web browser, checking the results, and then going back to Word. This hack will show you how to display the search results in Word's New Document Task Pane.

This hack uses a plain text *.ini* file to store data and some Visual Basic for Applications (VBA) code that also uses VBScript regular expressions.



This hack will work only with Word 2003 for Windows.

Using Google from Word requires a bit of setup, but once you've installed the appropriate tools, you can use Google from within any Word macro.

Install the Web Services Toolkit

First, install the free Microsoft Office 2003 Web Services Toolkit 2.01. Search for it on the Microsoft web site (<http://www.microsoft.com/downloads/>) or Google it.

Create a New Template

Next, create a new template to hold your Google-related macros. The Web Services Toolkit will create some code so that you can work with Google. A separate template will help you keep track of the code. Create a new, blank document and save it as a Document Template named *GoogleTools.dot*.

Install the Google Interface VBA Code

From your new *GoogleTools.dot* template, select Tools → Macro → Visual Basic Editor. The Web Services Toolkit will have added a new item called Web Service References on the Tools menu, as shown in [Figure 5-19](#).

Select Tools → Web Service References to display the dialog shown in [Figure 5-20](#). Enter *google* in the Keywords field and click the Search button. When the web service is found, check the box next to it, and click the Add button.

When you click the Add button, you'll notice a flurry of activity on your screen as the Web Services Toolkit installs several new class modules into your template project, as shown in [Figure 5-21](#).

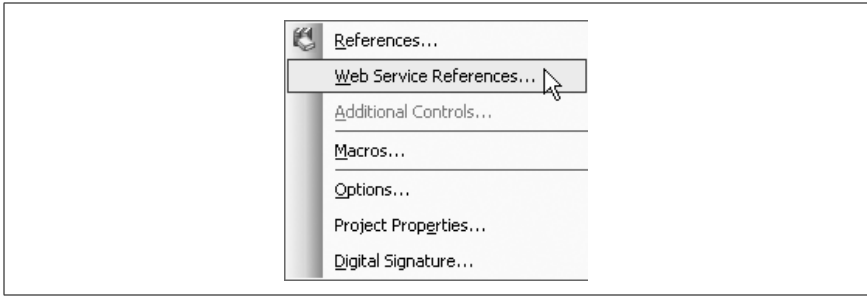


Figure 5-19. Creating a new reference for accessing Google

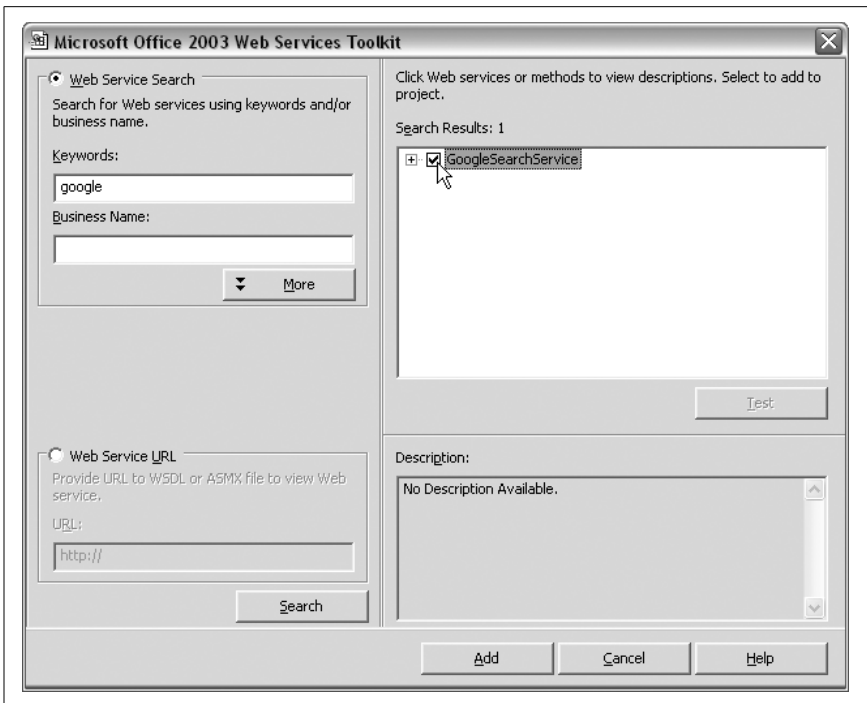


Figure 5-20. Locating the Google search web service



The Web Services Toolkit creates the code, but it actually comes from Google using Web Services Description Language (WSDL). The Toolkit interprets this information and generates the VBA code needed to access the web service—in this case, Google.

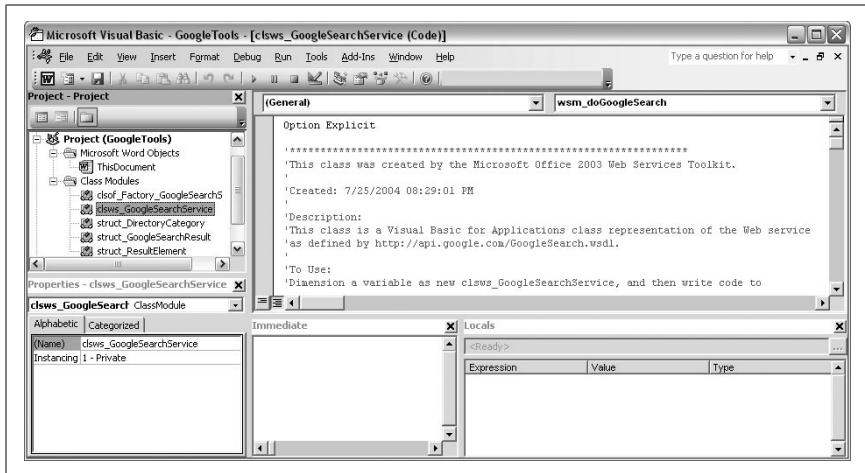


Figure 5-21. The code created by the Web Services Toolkit

The Code

With the *GoogleTools.dot* template you created open, select Tools → Macro → Macros and insert the following code, which consists of a procedure named *GoogleToTaskPane* and a supporting function named *StripHTML*.



Make sure you replace the value *insert key here* with your Google API developer's key.

```
Sub GoogleToTaskPane()
Dim vSearchResults As Variant
Dim v As Variant
Dim sResults As String
Dim sEntryName As String
Dim sEntryURL As String
Dim sLogFile As String
Dim sSearchDisplayTitle As String
Dim sSearchURL As String
Dim i As Integer

' Google API variables
Dim sGoogleAPIKey As String
Dim sSearchQuery As String
Dim lStart As Long
Dim lMaxResults As Long
Dim bFilter As Boolean
Dim sRestrict As String
Dim bSafeSearch As Boolean
Dim sLanguageRestrict As String
```

```

Dim sInputEncoding As String
Dim sOutputEncoding As String
Dim google_search As New clsWS_GoogleSearchService

' Initialize variables
sLogFile = "C:\google_taskpane.ini"
sGoogleAPIKey = "insert your key"
lStart = 1
lMaxResults = 10
bFilter = True
sRestrict = ""
bSafeSearch = False
sLanguageRestrict = ""
sInputEncoding = "UTF-8"
sOutputEncoding = "UTF-8"

' Hide the Task Pane
Application.CommandBars("Task Pane").Visible = False

' Remove existing items from New Document Task Pane
For i = 0 To 9
    sEntryURL = System.PrivateProfileString( _
        FileName:=sLogFile, _
        Section:="GoogleTaskPane", _
        Key:="URLName" & CStr(i))
    sEntryName = System.PrivateProfileString( _
        FileName:=sLogFile, _
        Section:="GoogleTaskPane", _
        Key:="EntryName" & CStr(i))
    If Len(sEntryURL) > 0 Then
        Application.NewDocument.Remove _
            FileName:=sEntryURL, _
            Section:=msoBottomSection, _
            DisplayName:=sEntryName, _
            Action:=msoOpenFile
    End If
Next i

' Get new search query
sSearchQuery = InputBox("Enter a Google query:")
If Len(sSearchQuery) = 0 Then Exit Sub

' Get search results
vSearchResults = google_search.wsm_doGoogleSearch( _
    str_key:=sGoogleAPIKey, _
    str_q:=sSearchQuery, _
    lng_start:=lStart, _
    lng_maxResults:=lMaxResults, _
    bln_filter:=bFilter, _
    str_restrict:=sRestrict, _
    bln_safeSearch:=bSafeSearch, _
    str_lr:=sLanguageRestrict, _
    str_ie:=sInputEncoding, _

```

```

        str_oe:=sOutputEncoding).resultElements

' Check for no results
On Error Resume Next
v = UBound(vSearchResults)
If Err.Number = 9 Then
    MsgBox "No results found"
    Exit Sub
ElseIf Err.Number <> 0 Then
    MsgBox "An error has occurred: " & _
        Err.Number & vbCrLf & _
        Err.Description
    Exit Sub
End If

' Add each result to the task pane
' and to the log file
i = 0
For Each v In vSearchResults
    sSearchURL = v.URL
    sSearchDisplayTitle = StripHTML(v.title)
    Application.NewDocument.Add _
        FileName:=sSearchURL, _
        Section:=msoBottomSection, _
        DisplayName:=sSearchDisplayTitle, _
        Action:=msoOpenFile

    System.PrivateProfileString( _
        FileName:=sLogFile, _
        Section:="GoogleTaskPane", _
        Key:="URLName" & CStr(i)) = sSearchURL
    System.PrivateProfileString( _
        FileName:=sLogFile, _
        Section:="GoogleTaskPane", _
        Key:="EntryName" & CStr(i)) = sSearchDisplayTitle
    i = i + 1
Next v

' Show the New Document Task Pane
CommandBars("Menu Bar").Controls("File").Controls("New...").Execute

End Sub

Function StripHTML(str As String) As String
Dim re As Object
Dim k As Long
On Error Resume Next
Set re = GetObject(Class:="VBScript.RegExp")
If Err.Number = 429 Then
    Set re = CreateObject(Class:="VBScript.RegExp")
    Err.Clear
ElseIf Err.Number <> 0 Then
    MsgBox Err.Number & vbCrLf & Err.Description
End If

```

```
' Check for common character entities by ASCII value
For k = 33 To 255
    re.Pattern = "&#" & k & ";"
    str = re.Replace(str, Chr$(k))
Next k

' Remove common HTML tags
re.Pattern = "<[>]+?>|&[^;]+?;"
re.Global = True
str = re.Replace(str, vbNullString)
StripHTML = str
End Function
```

This hack uses two parts of the Google search results: the URLs and titles. Google formats the search result title as HTML, but you can only put plain text in the Task Pane. The StripHTML function uses a few simple VBScript Regular Expressions to strip out common HTML tags (such as) and replace character entities (such as @) with their ASCII character equivalents.

It can be tricky to remove files from the Task Pane using VBA unless you know their exact name. This macro, however, stores the search results in a plain-text .ini file. The next time you do a search, you can easily remove the previous results. The macro uses a file named *C:\google_taskpane.ini*, which is defined in the GoogleToTaskPane procedure.

Running the Hack

After you insert the code, switch back to Word. Next, select Tools → Macro → Macros, choose GoogleToTaskPane, and click the Run button to display the dialog shown in [Figure 5-22](#).

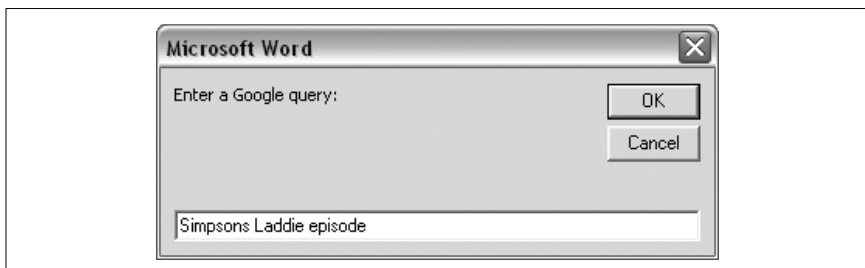


Figure 5-22. Entering a Google search that will display in the Task Pane

Enter your search terms and click the OK button. The New Document Task Pane appears and displays the search results, as shown in [Figure 5-23](#). Hover your mouse over any of the entries to display the URL. Click a URL to open the site in your web browser.

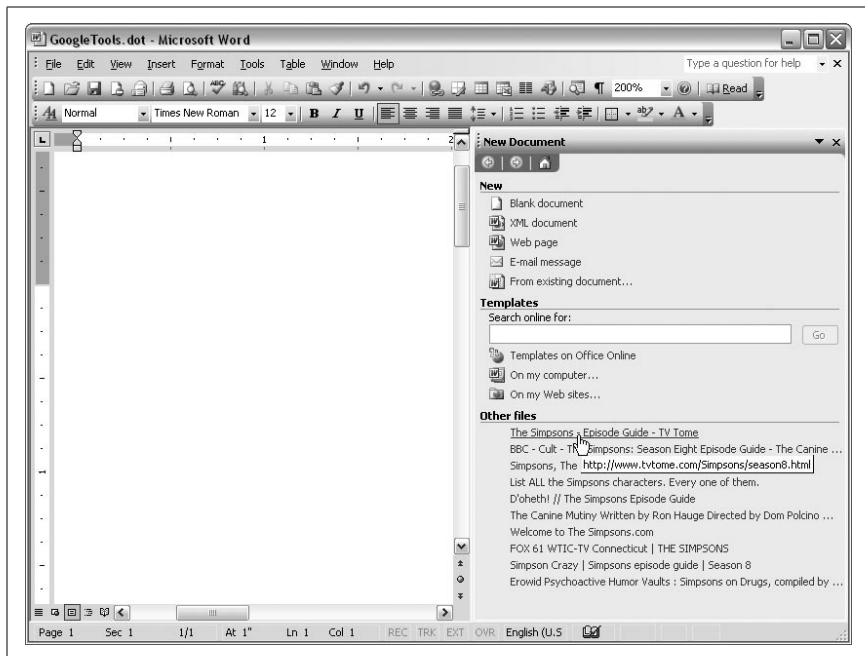


Figure 5-23. Google results displayed in the Task Pane

Every time you run a search, the macro removes the previous results from the Task Pane. If you want to remove the previous results without displaying new ones, click the Cancel button in the dialog box shown in Figure 5-22.



To make sure this handy macro loads automatically when Word starts, put *GoogleTools.dot* into your Startup folder, typically *C:\Documents and Settings\<username>\Application Data\Microsoft\Word\STARTUP*.

Hacking the Hack

To take this hack one step further, you can modify it to use the currently selected text as the search text, rather than displaying an input box and entering text.

The following macro, named *GoogleSelectionToTaskPane*, does a Google search of the currently selected text and displays the results in the Task Pane. The modified code is shown in bold.

```
Sub GoogleSelectionToTaskPane()  
    Dim vSearchResults As Variant  
    Dim v As Variant
```

```

Dim sResults As String
Dim sEntryName As String
Dim sEntryURL As String
Dim sLogFile As String
Dim sSearchDisplayTitle As String
Dim sSearchURL As String
Dim i As Integer

' Google API variables
Dim sGoogleAPIKey As String
Dim sSearchQuery As String
Dim lStart As Long
Dim lMaxResults As Long
Dim bFilter As Boolean
Dim sRestrict As String
Dim bSafeSearch As Boolean
Dim sLanguageRestrict As String
Dim sInputEncoding As String
Dim sOutputEncoding As String
Dim google_search As New clsws_GoogleSearchService

' Initialize variables
sLogFile = "C:\google_taskpane.ini"
sGoogleAPIKey = your_key_here
lStart = 1
lMaxResults = 10
bFilter = True
sRestrict = ""
bSafeSearch = False
sLanguageRestrict = ""
sInputEncoding = "UTF-8"
sOutputEncoding = "UTF-8"

' Hide the Task Pane
Application.CommandBars("Task Pane").Visible = False

' Remove existing items from New Document Task Pane
For i = 0 To 9
    sEntryURL = System.PrivateProfileString( _
        FileName:=sLogFile, _
        Section:="GoogleTaskPane", _
        Key:="URLName" & CStr(i))
    sEntryName = System.PrivateProfileString( _
        FileName:=sLogFile, _
        Section:="GoogleTaskPane", _
        Key:="EntryName" & CStr(i))
    If Len(sEntryURL) > 0 Then
        Application.NewDocument.Remove _
            FileName:=sEntryURL, _
            Section:=msoBottomSection, _
            DisplayName:=sEntryName, _
            Action:=msoOpenFile
    End If

```

```

Next i

' Move ends of selection to exclude spaces
' and paragraph marks
Selection.MoveStartWhile cset:=Chr$(32) & Chr$(19), _
    Count:=Selection.Characters.Count
Selection.MoveEndWhile cset:=Chr$(32) & Chr$(19), _
    Count:=-Selection.Characters.Count

' Get selection text for search
sSearchQuery = Selection.Text
If Len(sSearchQuery) = 0 Then Exit Sub

' Get search results
vSearchResults = google_search.wsm_doGoogleSearch( _
    str_key:=sGoogleAPIKey, _
    str_q:=sSearchQuery, _
    lng_start:=lStart, _
    lng_maxResults:=lMaxResults, _
    bln_filter:=bFilter, _
    str_restrict:=sRestrict, _
    bln_safeSearch:=bSafeSearch, _
    str_lr:=sLanguageRestrict, _
    str_ie:=sInputEncoding, _
    str_oe:=sOutputEncoding).resultElements

' Check for no results
On Error Resume Next
v = UBound(vSearchResults)
If Err.Number = 9 Then
    MsgBox "No results found"
    Exit Sub
ElseIf Err.Number <> 0 Then
    MsgBox "An error has occurred: " & _
        Err.Number & vbCrLf & _
        Err.Description
    Exit Sub
End If

' Add each result to the task pane
' and to the log file
i = 0
For Each v In vSearchResults
    sSearchURL = v.URL
    sSearchDisplayTitle = StripHTML(v.title)
    Application.NewDocument.Add _
        FileName:=sSearchURL, _
        Section:=msoBottomSection, _
        DisplayName:=sSearchDisplayTitle, _
        Action:=msoOpenFile

    System.PrivateProfileString( _
        FileName:=sLogFile, _

```

```

        Section:="GoogleTaskPane", _
        Key:="URLName" & CStr(i)) = sSearchURL
System.PrivateProfileString( _
    FileName:=sLogFile, _
    Section:="GoogleTaskPane", _
    Key:="EntryName" & CStr(i)) = sSearchDisplayTitle
    i = i + 1
Next v

' Show the New Document Task Pane
CommandBars("Menu Bar").Controls("File").Controls("New...").Execute

End Sub

```

To help ensure a good Google search, the following two lines collapse two ends of the selection if they contain spaces or a paragraph mark:

```

Selection.MoveStartWhile cset:=Chr$(32) & Chr$(19), _
    Count:=Selection.Characters.Count
Selection.MoveEndWhile cset:=Chr$(32) & Chr$(19), _
    Count:=-Selection.Characters.Count

```

—Andrew Savikas