# Windows Automation

Using Ruby to create an automation controller to issue commands and queries to an automation server

# It’s not often that you hear Microsoft and Innovation in the same sentence.

- Microsoft is usually thought of as the acquirer or copier of innovation but there are some areas that they have pioneered such as …
  - Cross Platform Operating System that once supported the PowerPC, DEC Alpha, MIPS R4000, etc.
  - ODBC (Open Database Connectivity)
  - DDE (Dynamic Data Exchange)
  - OLE (Object Linking and Embedding)
  - Microsoft BOB
  - Windows Automation

# Windows Automation …

- Is a technique to allow a developer to control another application by using a readily available Windows protocol.
- Is based on a subset of Component Object Model.
- Is built into many commercial and some open source products.
- Has been available since Win95.
- Has been renamed Automation

# Put Another Way…

- Automation is an openly defined protocol to allow an application (the server) to expose some or all of it's methods and properties for the purpose of being executed by an external program (the client).
- By way of example: Microsoft Word is an automation server. The program that a developer writes to interact with Word is the client.

# Automation is available for …

- All Microsoft Office products
- Internet Explorer
- Open Office Suite of programs
- Lotus Notes
- Adobe Acrobat
- Lotus 1-2-3
- AutoCAD
- Oracle
- SAS

# Automation is not available for

- Google Chrome
- Mozilla Firefox
- Safari
- Microsoft Calc
- Microsoft Solitaire
- Microsoft Notepad
- Microsoft Paint

# Ruby Implementation

- The Ruby implementation to create an automation client is provided by the “win32ole” extension that comes standard with the install for Windows.

# Initial Steps …

- Ensure that the application that you want to control is installed on your computer.
- Place, require “win32ole” in your Ruby source code.
- Further, you need to know the ProgID associated with the program that you want to control.

# Some Server / ProgID’s

| **Application (Server)** | **ProgID** |
|---|---|
| Excel | Excel.Application |
| Word | Word.Application |
| Internet Explorer | InternetExplorer.Application |
| Lotus Notes | Notes.NotesSession |
| SAS | SAS.Application |
| OpenOffice | com.sun.star.frame.Desktop |
| AutoCAD | AutoCAD.Application |
| Oracle | OracleInProcServer.XOraSession |

# Initial Steps (continued) …

- You must create an object that represents the automation client to the program that you wish to control.
- To do this in Ruby simply create a new instance of the WIN32OLE object with the ProgID as the constructor parameter and assign it to a variable.
- Let’s start with something simple, we’ll control Internet Explorer:

```
ie = WIN32OLE.new(“InternetExplorer.Application”)
```

# Initial Steps (continued) …

- But why doesn’t Internet Explorer appear on the screen? Because there is a visible property that in most cases defaults to false. Set it to true to see the new instance of the browser.
- Something to keep in mind is that the WIN32OLE object has methods and properties defined, as with any other Ruby object. However, it does not have specific methods and properties to control the specific automation server you wish to control. It merely acts as the conduit to dispatch the method called to its automation server.

# Initial Steps (continued) …

- Next, issue the command to go to the home page defined in the browser:

  ```
  ie.gohome
  ```

- Next, navigate to Google:

  ```
  ie.navigate("www.google.com")
  ```

- Now adjust the height of the browser window

  ```
  ie.height = 300
  ```

- Now adjust the width of the browser window

  ```
  ie.width = 640
  ```

- Now close the browser

  ```
  ie.close
  ```

- Oops! You'll get the NoMethodError. This is because if the method you call on the object is not defined in the WIN32OLE object it is sent to the automation server for completion. If the server does not expose that method an error is returned. In order to close the browser issue:

  ```
  ie.quit
  ```

# Something more complicated …

- Goal: to launch Excel and place two numbers in cells on a worksheet then add those two numbers together and place in a third cell.
- First, create the Excel automation object:

```
excel=WIN32OLE.new(“Excel.Application”)
excel.visible=true
```

- Next, add a workbook and place two numbers in two cells:

```
excel.Workbooks.Add
excel.Range(“a1”).value=100
excel.Range(“a2”).value=200
```

- Last, add those two numbers together and place in a third cell:

```
excel.Range(“a3”).formula=“=a1+a2”
```

# Next Example: Excel and Word

- Goal: Create an Excel spreadsheet and fill a column with words then move those to a Word document.
- First, create a Excel automation object:

```
excel=WIN32OLE.new(“Excel.Application”)
excel.visible=true
exce.workbooks.add
```

- Then, create a Word automation object:

```
word=WIN32OLE.new(“Word.Application”)
word.visible=true
word.documents.add
```

- Next, create a words array and move it to the Excel spreadsheet:

```
color_code=[“Bad”,”Boys”,”Ruin”,”Our”,”Young”, “Grass”,”But”, “Violets”, “Grow”,
    “Wildly”]
excel.Range(“A1:A10”).each do |cell|
    cell.value=color_code[cell.row - 1]
end
```

# Example:Excel and Word (Cont’d)…

- Lastly, move the Excel data to the Word document:

```
excel.Range(“A1:A10”).each do |cell|
    word.Selection.TypeText(cell.text)
    word.Selection.TypeText(“ “)
end
```

# Final Example: Excel, Word, Lotus Notes

- Goal: to connect to a Lotus Notes database, read some items out of documents, move those items to Excel and then subsequently move those to Word.
- Lotus Notes is a noSQL database that was first released in 1989. It's hierarchy is Lotus Notes Application > Session > Database > Documents and Views > Items within Documents
- In this example there will be one item called "Words" that we will extract from each document.

# Final Example: Excel, Word, Lotus Notes (cont’d) …

□ First, connect to Lotus Notes with a session object

```
Notes=WIN32OLE.new(“Notes.NotesSession”)
```

□ Next, instantiate an Excel and Word object

```
excel=WIN32OLE.new(“Excel.Application”)
excel.visible=true
exce.workbooks.add
word=WIN32OLE.new(“Word.Application”)
word.visible=true
word.documents.add
```

# Final Example: Excel, Word, Lotus Notes (cont’d) …

- Next, get a notes database object

```
db = notes.getDatabase(“”, “Ode to 1918
    Pandemic.nsf”)
```

- Next, from the database object get the All view

```
view = db.getView(“All”)
```

- For this example we are going to need to know how many documents we are going to read.  The view object has a property for that

```
number_documents = view.entrycount
```

# Final Example: Excel, Word, Lotus Notes (cont'd) …

- Create an Excel Range literal that will represent the range that we will [lace the words extracted from the database

```
excel_range =
  "A1:A#{number_documents.to_s}"
```

- Next, get the first notes document in the All view

```
doc = view.getFirstDocument
```

# Final Example: Excel, Word, Lotus Notes (cont’d) …

- Then, iterate through the excel range extracting the appropriate item from the document pointed to by the document object

```
excel.range(excel_range).each do |cell|
    cell.value = doc.getFirstItem(“Words”).text
    doc = view.getNextDocument(doc)
end
```

# Final Example: Excel, Word, Lotus Notes (cont’d) …

- Then, iterate over the excel_range and copy those words to the Word document to form sentences and paragraphs

```
excel.Range(“A1:A10”).each do |cell|
    if cell.text == “<enter>” then
         word.Selection.TypeParagraph()
    else
         word.Selection.TypeText(cell.text)
         word.Selection.TypeText(“ “)
    end
end
```

# Conclusion

- The Ruby WIN32OLE extension can be used to create programs that control other applications with Windows Automation.

# Appendix A: Resources

- □ Microsoft Office Products … refer to Appendix B
- □ Internet Explorer Objects and Properties can be found at:
  - https://msdn.microsoft.com/en-us/library/aa752084%28v=vs.85%29.aspx
- □ Open Office Products:
  - https://www.openoffice.org/udk/common/man/tutorial/office_automation.html
- □ Lotus Notes:
  - http://www-01.ibm.com/support/knowledgecenter/SSVRGU_9.0.1/com.ibm.designer.domino.main.doc/H_LOTUSSCRIPT_NOTES_CLASSES.html

# Appendix A: Resources (cont’d)

- Adobe Acrobat:

  http://www.adobe.com/devnet/acrobat/interapplication_communication.html

- Lotus 1-2-3:

  ftp://ftp.software.ibm.com/pub/lotusweb/ssdev/Getmost.pdf

- Autocad-I wasn’t able to find a central repository for a resource but the following link will show an example:

  https://groups.google.com/forum/#!topic/borland.public.delphi.oleautomation/hWHV2y-MbN8

# Appendix A: Resources (cont’d)

- □ Oracle:

  http://www.oracle.com/technetwork/database/windows/index-088441.html

- □ SAS:

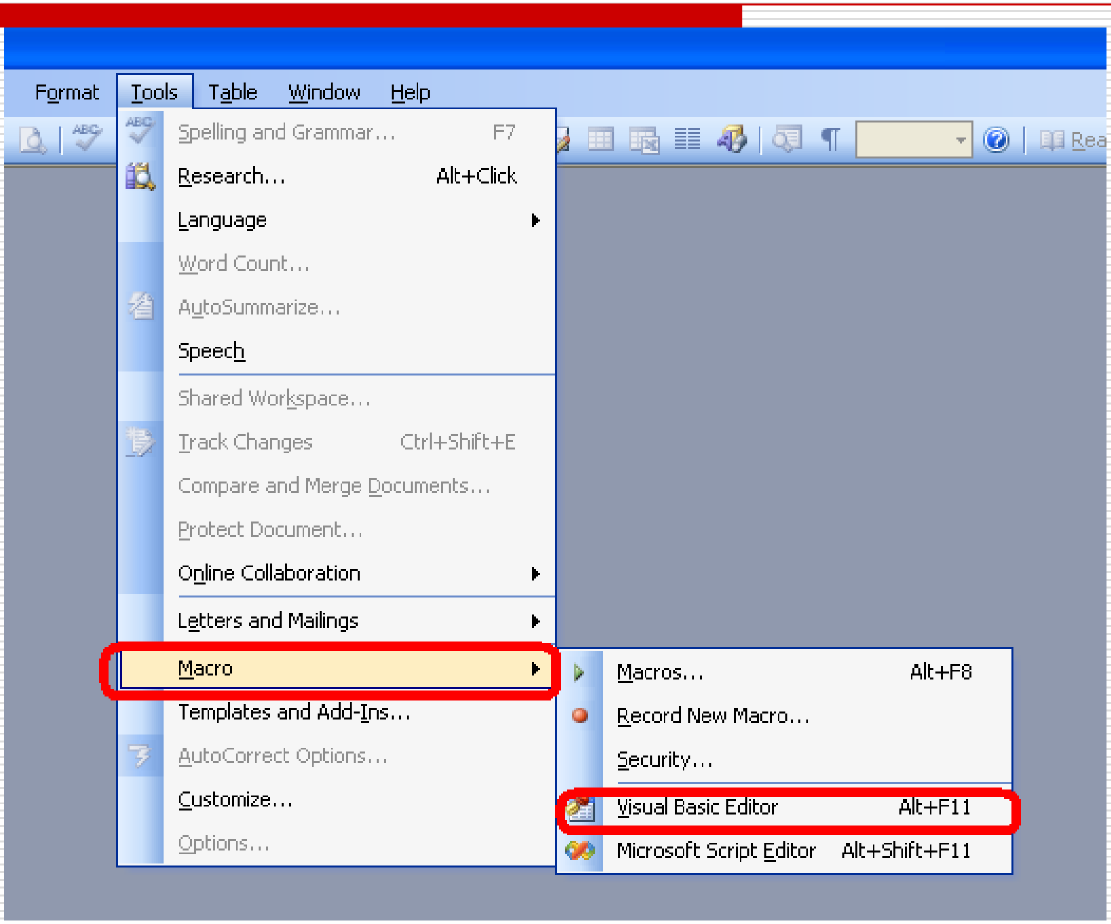
  http://support.sas.com/documentation/cdl/en/hostwin/63285/HTML/default/viewer.htm#olecreateinst.htm

  http://www2.sas.com/proceedings/sugi22/ADVTUTOR/PAPER46.PDF

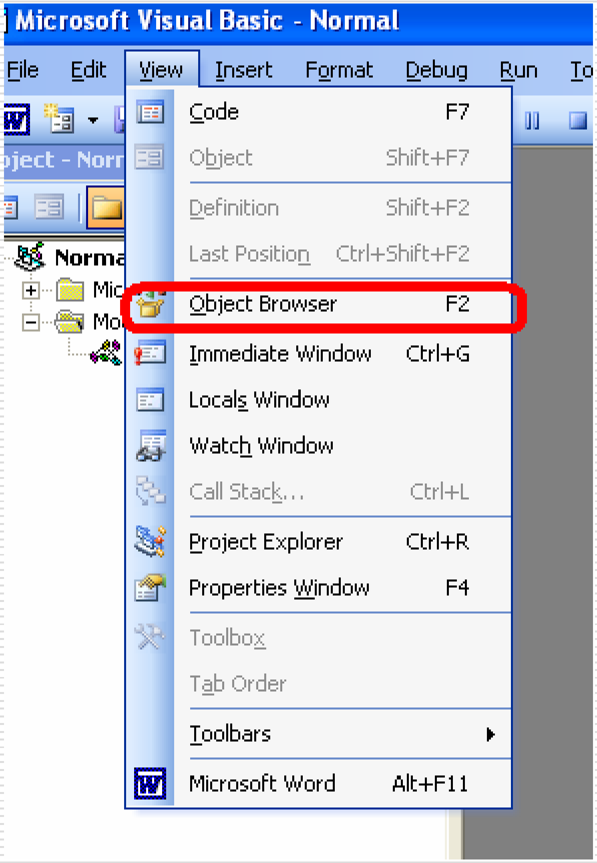
# Appendix B: How to find methods and properties with Microsoft Products

- Most, if not all Microsoft Office products support macros using VBA.
- To find out all the methods and properties for Microsoft Office software enter the Visual Basic area of the product that you are interests you.  In the examples shown in this presentation I have shown Windows Automation with Microsoft Word 2003.  In that version you may enter VBA from the main menu and pressing Tools > Macro > Visual Basic.  Consult the help file of your Microsoft software to enter VBA from your version.  The following slides show how to find which methods and properties are exposed.

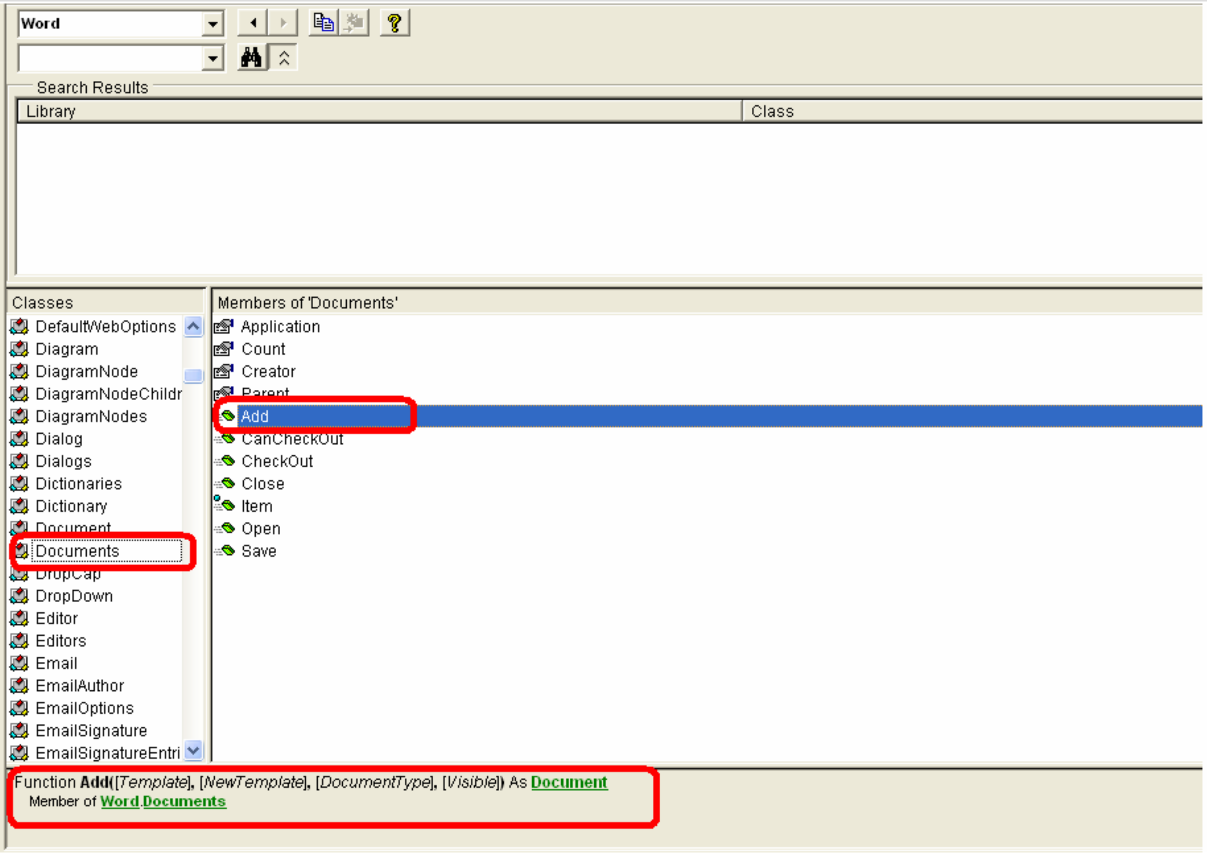
# Appendix B (cont'd) … First, on the menu press Tools > Macro > Visual Basic Editor



# Appendix B (cont'd) … Second, on the menu press View > Object Browser



Appendix B (cont'd) … Classes will be shown on the left list and methods and properties of that class are on the right list. Select the desired method or property and definitions are shown in the panel below the lists.



Word

Classes:
DefaultWebOptions
Diagram
DiagramNode
DiagramNodeChildr
DiagramNodes
Dialog
Dialogs
Dictionaries
Dictionary
Document
Documents
DropCap
DropDown
Editor
Editors
Email
EmailAuthor
EmailOptions
EmailSignature
EmailSignatureEntri

Members of 'Documents':
Application
Count
Creator
Parent
Add
CanCheckOut
CheckOut
Close
Item
Open
Save

```
Function Add([Template], [NewTemplate], [DocumentType], [Visible]) As Document
  Member of Word.Documents
```