Following is a description of the three programs created in this Final VBA project.

FINAL VBA PROJECT

Chris Ellis

2.1 Executive summary of the project

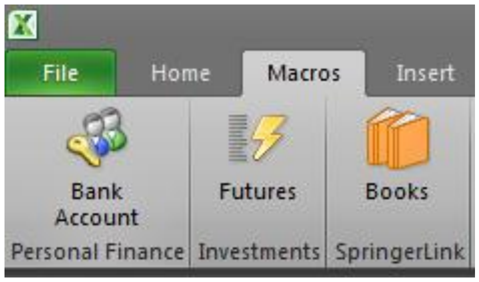
As a frugal saver and rate shopper, I have opened accounts at various financial institutions. The time it takes to keep track of the various passwords and to login to the accounts is time consuming and annoying when you could be doing something better like watching BYU football or the Jimmer show. This project attempts to alleviate that unneeded stress by automating the process of going to the account and retrieving the current balance in the case of banking and brokerage account, and in the case of the credit card accounts it retrieves the most recent statement's balance. This program interacts with American Express, ING Direct, Vanguard, Zions, iGO banking, Bank of America (credit card), Scottrade, and the 401(k) through Wells Fargo. The functionality makes this process seamless so that the user can do all of this by just the click of the button, no more entering passwords and user IDs.

In addition to the macros created to serve some financial purposes, another macro was created to aid in downloading books through the SpringerLink database offered at BYU. Once the URL of where the book's webpage is copied (Ctrl + c), then the user initiates the command and follows the prompts to enter the URL, net ID, and password enabling the macro to proceed in downloading all of the chapters and naming them appropriately, all while numbering the chapters to help keep them in order.

Just on the side, there is also a macro that retrieves futures information of the S&P 500 front month beginning with April of 1982 and ending with the most recent closed day.

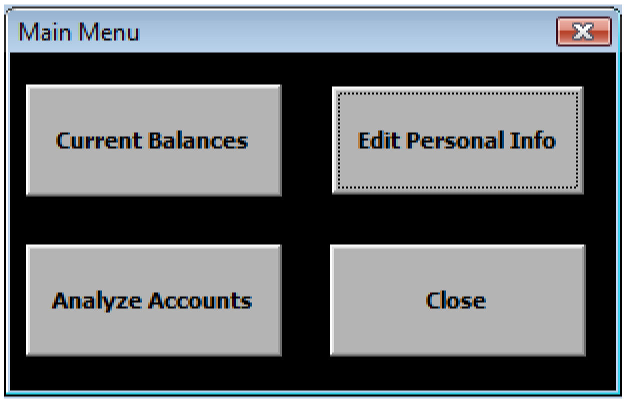
2.2 Implementation documentation

First, in order to access the macros, the ribbon in Excel has been modified to enable the access of the macros:

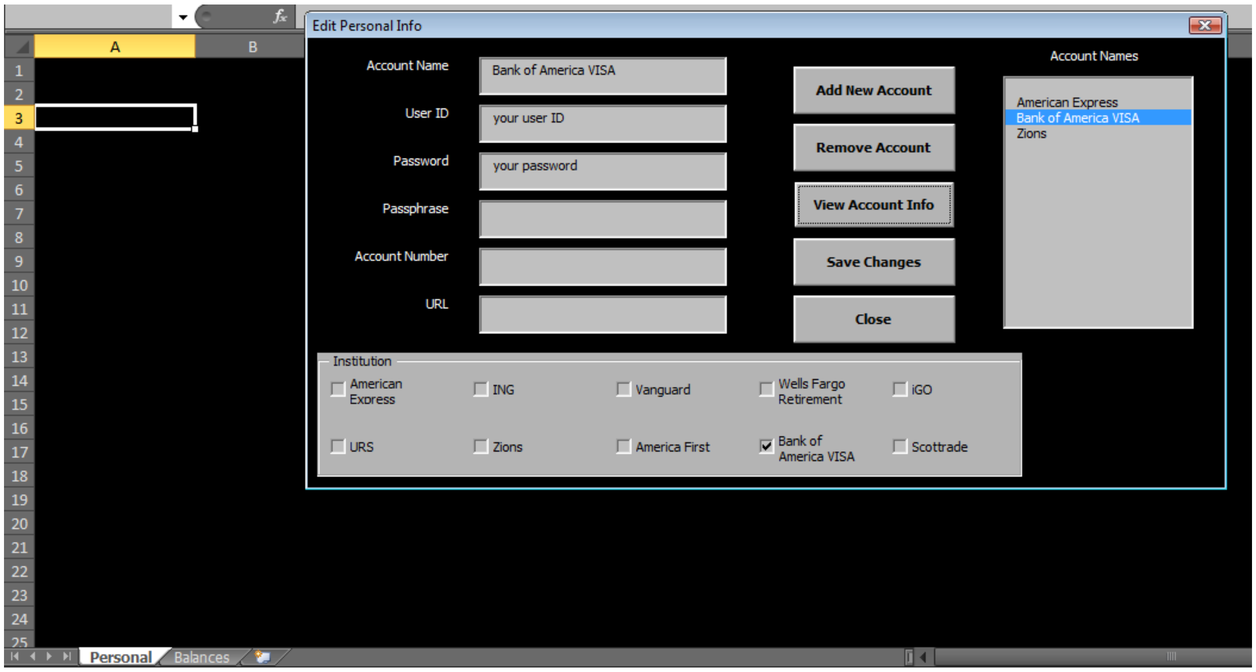


These macros are accessible through a new tab called Macros that is tied only to that workbook.

Once, one clicks on the Bank Account image, the macro begins by pulling up the following userform:



From here, four options are given. If the Edit Personal Info button is selected, then the spreadsheet containing the confidential information along with a userform to modify its contents appears in order to facilitate an easy manner in which one can keep track of that information. This spreadsheet that appears is only visible through a command given through VBA. With the userform, the operator can view, add, remove, and modify account information that is necessary to access the particular financial institution account that is held.



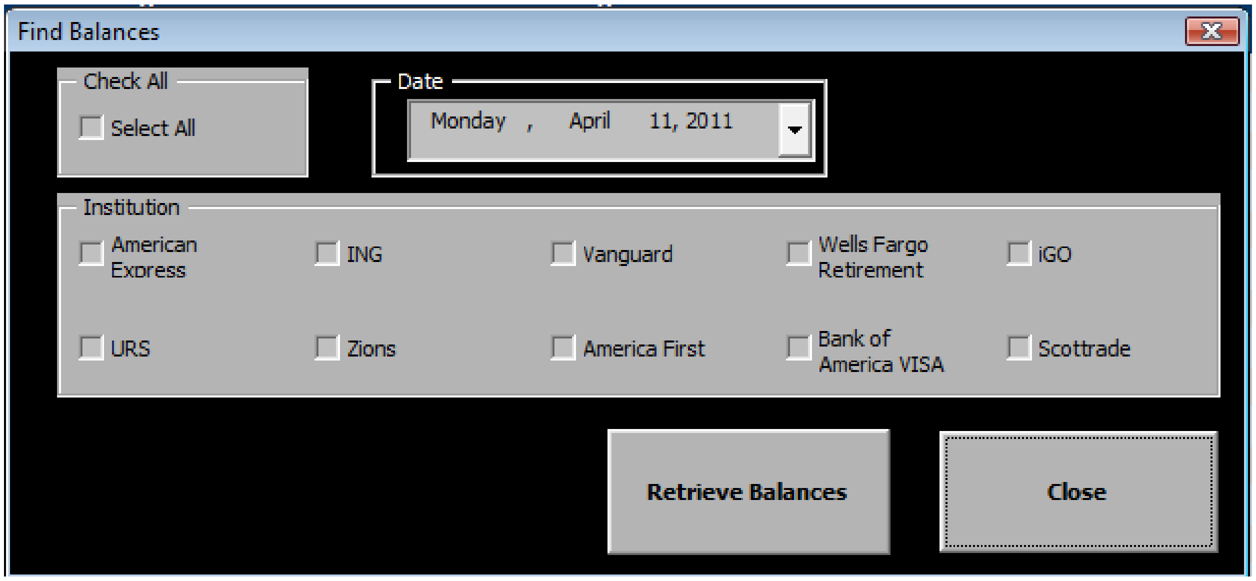
Account Name: Bank of America VISA
User ID: your user ID
Password: your password
Passphrase:
Account Number:
URL:

Account Names:
American Express
Bank of America VISA
Zions

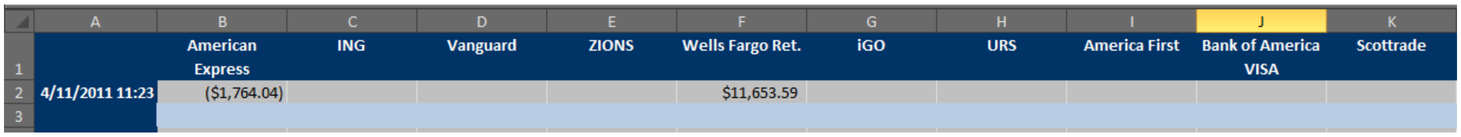
Institution:

☐ American Express	☐ ING	☐ Vanguard	☐ Wells Fargo Retirement	☐ iGO
☐ URS	☐ Zions	☐ America First	☑ Bank of America VISA	☐ Scottrade

If Current Balances button is selected then another form appears:

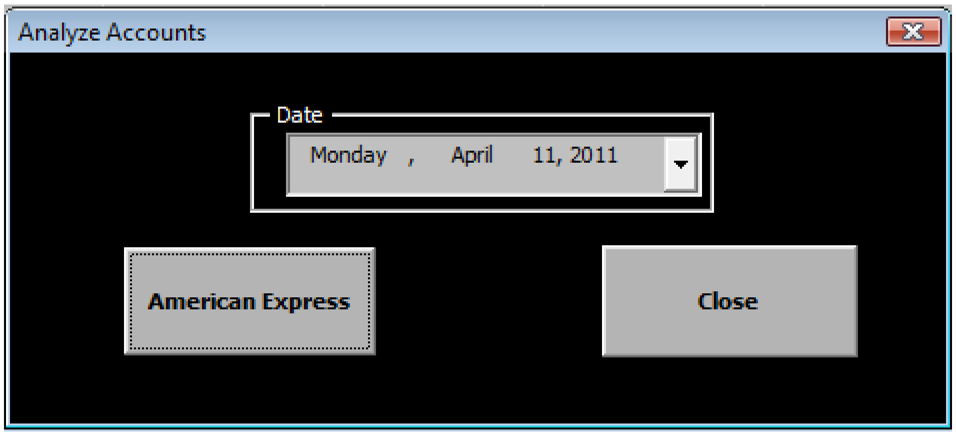


Here, the user can select the current balances of the accounts that one wants to retrieve, which will then be populated into this spreadsheet with a timestamp of when this occurred, for example:



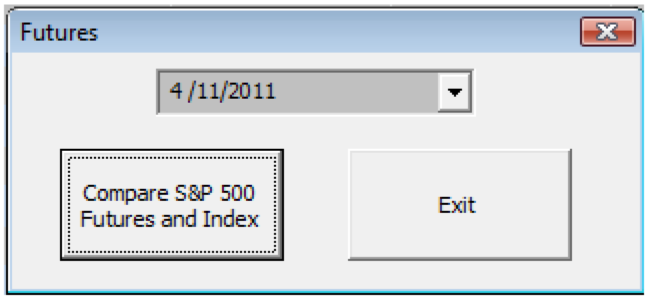
	A	B	C	D	E	F	G	H	I	J	K
1		American Express	ING	Vanguard	ZIONS	Wells Fargo Ret.	iGO	URS	America First	Bank of America VISA	Scottrade
2	4/11/2011 11:23	($1,764.04)				$11,653.59					
3											

If the Analyze Accounts button is selected then this userform appears:

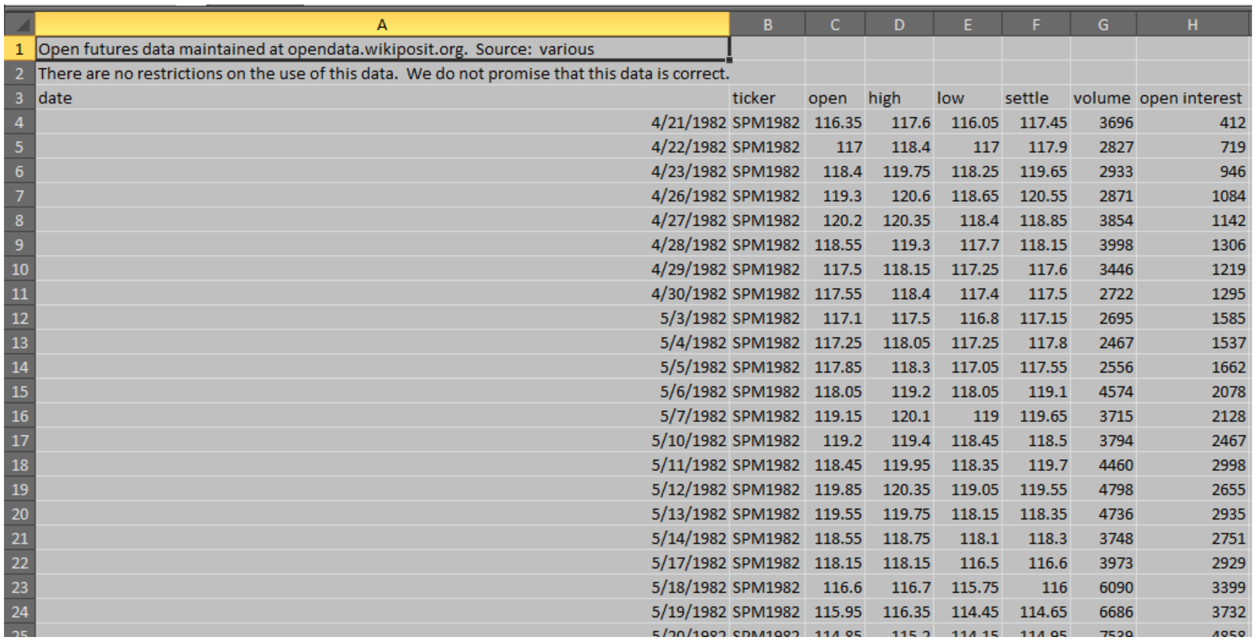


Currently in this base model, only American Express has some analytical abilities, which is limited to just presenting the user a summary of the most recent statement in spreadsheet form.

Regarding when the Futures macro is selected, then this userform appears:



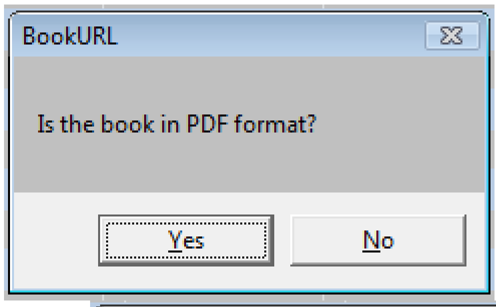
Upon selecting the command to compare the S&P 500 futures, the information is retrieved and populated into a spreadsheet with information beginning from 1982:

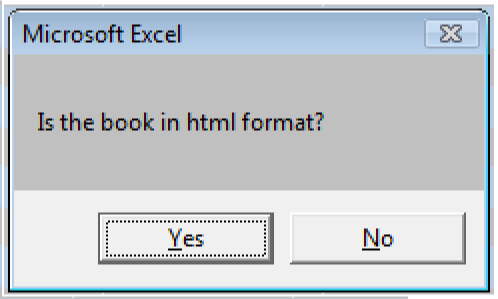


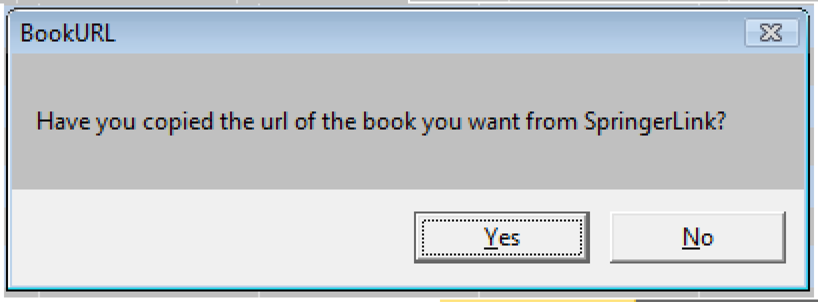
	A	B	C	D	E	F	G	H
1	Open futures data maintained at opendata.wikiposit.org. Source: various							
2	There are no restrictions on the use of this data. We do not promise that this data is correct.							
3	date	ticker	open	high	low	settle	volume	open interest
4	4/21/1982	SPM1982	116.35	117.6	116.05	117.45	3696	412
5	4/22/1982	SPM1982	117	118.4	117	117.9	2827	719
6	4/23/1982	SPM1982	118.4	119.75	118.25	119.65	2933	946
7	4/26/1982	SPM1982	119.3	120.6	118.65	120.55	2871	1084
8	4/27/1982	SPM1982	120.2	120.35	118.4	118.85	3854	1142
9	4/28/1982	SPM1982	118.55	119.3	117.7	118.15	3998	1306
10	4/29/1982	SPM1982	117.5	118.15	117.25	117.6	3446	1219
11	4/30/1982	SPM1982	117.55	118.4	117.4	117.5	2722	1295
12	5/3/1982	SPM1982	117.1	117.5	116.8	117.15	2695	1585
13	5/4/1982	SPM1982	117.25	118.05	117.25	117.8	2467	1537
14	5/5/1982	SPM1982	117.85	118.3	117.05	117.55	2556	1662
15	5/6/1982	SPM1982	118.05	119.2	118.05	119.1	4574	2078
16	5/7/1982	SPM1982	119.15	120.1	119	119.65	3715	2128
17	5/10/1982	SPM1982	119.2	119.4	118.45	118.5	3794	2467
18	5/11/1982	SPM1982	118.45	119.95	118.35	119.7	4460	2998
19	5/12/1982	SPM1982	119.85	120.35	119.05	119.55	4798	2655
20	5/13/1982	SPM1982	119.55	119.75	118.15	118.35	4736	2935
21	5/14/1982	SPM1982	118.55	118.75	118.1	118.3	3748	2751
22	5/17/1982	SPM1982	118.15	118.15	116.5	116.6	3973	2929
23	5/18/1982	SPM1982	116.6	116.7	115.75	116	6090	3399
24	5/19/1982	SPM1982	115.95	116.35	114.45	114.65	6686	3732

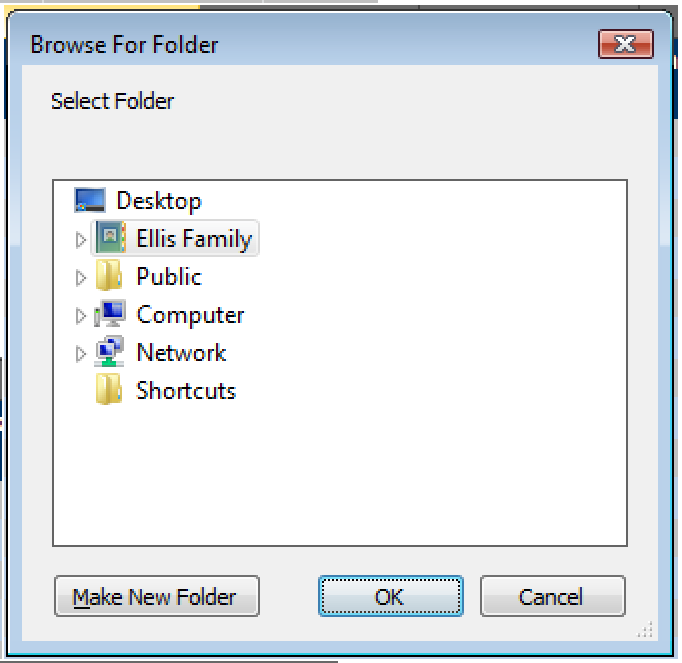
As more than 7,000 rows of data are populated into this spreadsheet, it takes some time for it to complete its task.

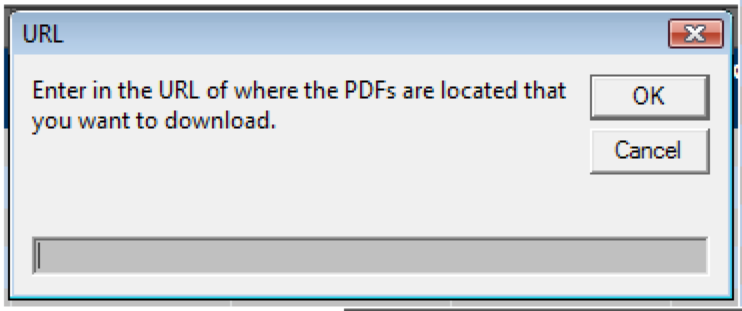
When the Books macro is selected, several pieces of information are needed from the user. The macro needs to know if the file to be downloaded is in the format of html or pdf, the URL of where the book is in SpringerLink, the net ID, and password so that the account can be accessed. This information obtained using input and message boxes:

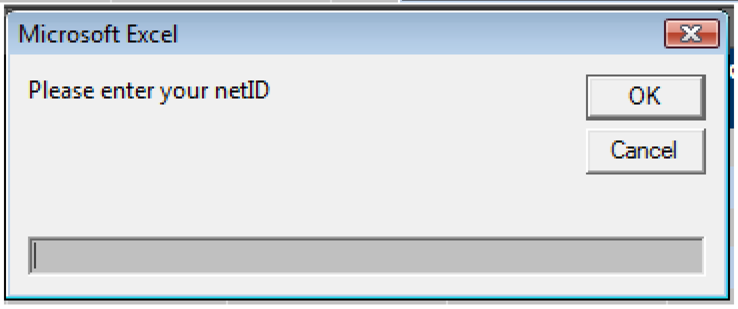

BookURL
Is the book in PDF format?
Yes
No

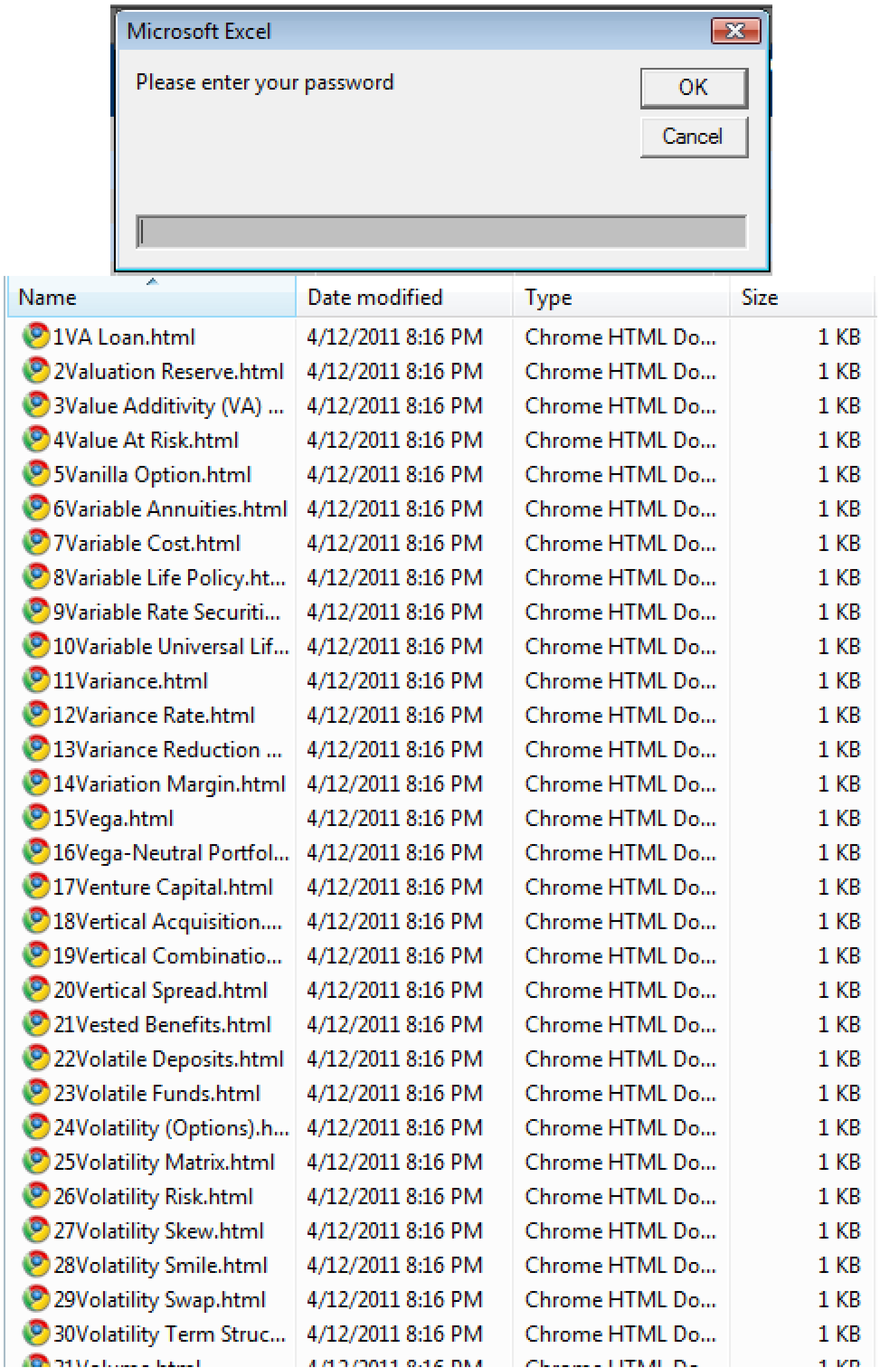

Microsoft Excel
Is the book in html format?
Yes
No


BookURL
Have you copied the url of the book you want from SpringerLink?
Yes
No


Browse For Folder
Select Folder
Desktop
Ellis Family
Public
Computer
Network
Shortcuts
Make New Folder
OK
Cancel


URL
Enter in the URL of where the PDFs are located that you want to download.
OK
Cancel


Microsoft Excel
Please enter your netID
OK
Cancel



Microsoft Excel

Please enter your password

Name	Date modified	Type	Size
1VA Loan.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
2Valuation Reserve.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
3Value Additivity (VA) ...	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
4Value At Risk.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
5Vanilla Option.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
6Variable Annuities.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
7Variable Cost.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
8Variable Life Policy.ht...	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
9Variable Rate Securiti...	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
10Variable Universal Lif...	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
11Variance.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
12Variance Rate.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
13Variance Reduction ...	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
14Variation Margin.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
15Vega.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
16Vega-Neutral Portfol...	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
17Venture Capital.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
18Vertical Acquisition....	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
19Vertical Combinatio...	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
20Vertical Spread.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
21Vested Benefits.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
22Volatile Deposits.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
23Volatile Funds.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
24Volatility (Options).h...	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
25Volatility Matrix.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
26Volatility Risk.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
27Volatility Skew.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
28Volatility Smile.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
29Volatility Swap.html	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB
30Volatility Term Struc...	4/12/2011 8:16 PM	Chrome HTML Do...	1 KB

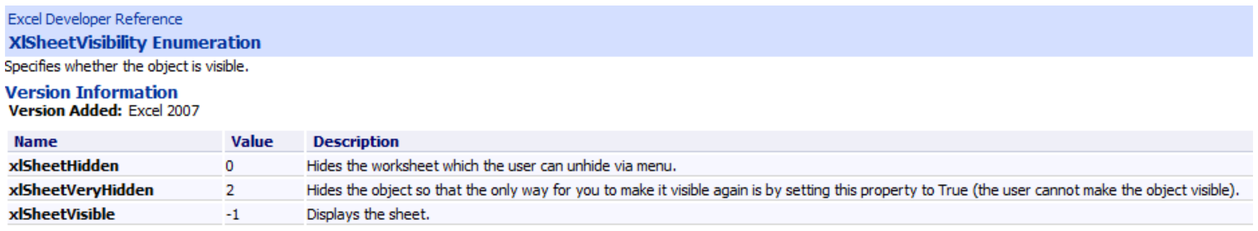
Figure 1

From there, the macro takes this information and navigates itself to the webpage where the links for the book's chapters are held. Then, the files are downloaded in the order given from the book into the folder of the user's selection. I find this to very helpful so that one doesn't have to download chapter by chapter. For example, with the Encyclopedia of Finance, there are probably more than a couple thousand files and if a user wanted to have access to that on their computer that would take a painstaking amount of time; whereas, the macro doesn't care, it just takes care of it. In Figure 1, an example of just the entries under V in the Encyclopedia of Finance.

2.3 Discussion of learning and conceptual difficulties encountered.

Much was learned in completing this project.

Given that every account requires a password and user ID, it can be difficult to keep track of this information in a secure manner. Many people keep this sort of information on a spreadsheet or on some piece of paper. This project assumes that the user will keep it in this workbook and in a particular spreadsheet within the workbook. In order to add another layer of safety with this type of sensitive information, the passwords and user IDs were stored on a spreadsheet whose visible property is set to xlSheetVeryHidden. There are three properties under visible:



Excel Developer Reference

XlSheetVisibility Enumeration

Specifies whether the object is visible.

Version Information

Version Added: Excel 2007

Name	Value	Description
xlSheetHidden	0	Hides the worksheet which the user can unhide via menu.
xlSheetVeryHidden	2	Hides the object so that the only way for you to make it visible again is by setting this property to True (the user cannot make the object visible).
xlSheetVisible	-1	Displays the sheet.

The xlSheetHidden is probably the property that most Excel users are familiar with when it comes to hiding worksheets; the xlSheetVeryHidden is only made visible when the property is set to True, thus requiring VBA to be executed. This isn't something that could be unhidden through some sort of menu command. If one were to password protect the workbook then another layer of security would be added, sufficient at least to prevent the novice from obtaining confidential information.

Accessing the balance information from the various websites held by financial institutions proved to be the most difficult task by far. Each website is different and some of them were very different. Mastering VBA sufficiently to manipulate the html in order to provide the desired information was what I would consider the greatest achievement through this project.

Just to share some of this experience, let me start with iGo. Like many of the banks that I encountered, there is more than one webpage on which one can enter the user ID and password in order to get in the account, and some webpages are certainly more difficult to work with than others are. With iGo, after entering the information and arriving to the webpage where the balance information was held, at first, I just could not get the webpage imported into Excel. Every time I tried to do so, the spreadsheet would just be empty and that is when I learned about frames. You might ask what frames are and that may be because most websites don't use them; they are somewhat confusing at best. It is when the html programmer decided to have more than one webpage on the same page. I don't why it was done that way and why the information isn't just on one page; I just know that after going through the html code and deciphering what was going on through my limited scope of understanding about html that there was another webpage that could be opened to acquire the desired data.

I also found out that some webpages have forms. Knowing that there are forms on the page are important because the form has to be referenced in the VBA code in order for it be found. Sometimes this was problematic just because it required a lot of hunting through the html code in order to get what was wanted. Even more problematic was that sometimes in order to login through the form, Java had to be executed. Given that VBA doesn't work with Java, some ingenuity and luck had to take place. For example, with ING Direct, just to enter in the pin through VBA, I determined that the keyboard option would have be made available instead of using a mouse to point on a particular image that would then execute some code in Java. Just to make that keyboard on the webpage option available, I developed the code that would go through the different tags in html and then find the right one and click on it:

```
Dim oneTag As Variant
Dim x As Long
Dim tagname As String
For x = 1 To agent1.explorer.document.body.all.Length
  tagname = ""
  On Error Resume Next
    tagname = agent1.explorer.document.body.all(x).tagname


  On Error GoTo 0


  If tagname = "A" Then
    If Right(agent1.explorer.document.body.all(x).innerhtml, 8) = "keyboard" Then
       agent1.explorer.document.body.all(x).Click
      If Right(agent1.explorer.document.body.all(x).innerhtml, 8) = "keyboard" _
      Then Exit For
    End If
  End If
Next
```

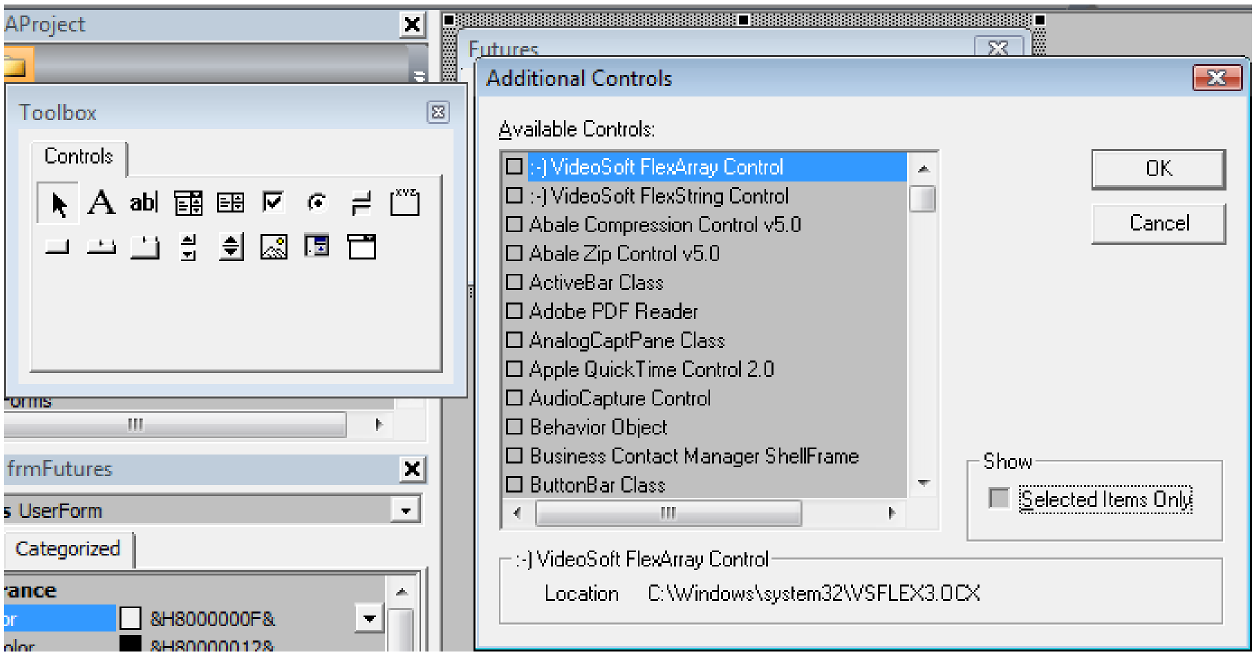
Here is a similar piece of code that had to go through the html just to get to the page where the password could be entered:

```
Dim oneTag As Variant
Dim x As Long
Dim tagname As String
For x = 1 To agent1.explorer.document.body.all.Length
  tagname = ""
  On Error Resume Next
    tagname = agent1.explorer.document.body.all(x).tagname
    If tagname = "A" Then Debug.Print x, tagname, _
    Left(Right(agent1.explorer.document.body.all(x).innerhtml, 13), 7)
  On Error GoTo 0


  If tagname = "A" Then
    If Left(Right(agent1.explorer.document.body.all(x).innerhtml, 13), 7) = "Sign In" Then
       agent1.explorer.document.body.all(x).Click
      If Left(Right(agent1.explorer.document.body.all(x).innerhtml, 13), 7) = "Sign In" Then Exit For
    End If
  End If
Next
```

I found that once the data was imported into a spreadsheet that using the ability to record macros was very helpful as I developed this code to scan the imported data provided from Wells Fargo:

```
Cells.find(What:="Vested Balance", After:=activecell, _
        LookIn:=xlFormulas, LookAt:=xlPart, SearchOrder:=xlByRows, _
        SearchDirection:=xlNext, MatchCase:=False, SearchFormat:=False).Activate

Application.DisplayAlerts = False

Selection.TextToColumns Destination:=activecell, DataType:=xlDelimited, _
        TextQualifier:=xlDoubleQuote, ConsecutiveDelimiter:=False, Tab:=False, _
        Semicolon:=False, Comma:=False, Space:=False, Other:=True, OtherChar _
        :="$", FieldInfo:=Array(Array(1, 1), Array(2, 1), Array(3, 1), Array(4, 1)), _
        TrailingMinusNumbers:=True

Cells.find(What:="Vested Balance", After:=activecell, LookIn:=xlFormulas _
        , LookAt:=xlPart, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
        MatchCase:=False, SearchFormat:=False).Activate

activecell.Offset(0, 1).Activate

Selection.TextToColumns Destination:=activecell, DataType:=xlDelimited, _
        TextQualifier:=xlDoubleQuote, ConsecutiveDelimiter:=True, Tab:=False, _
        Semicolon:=False, Comma:=False, Space:=True, Other:=False, OtherChar _
        :="$", FieldInfo:=Array(Array(1, 1), Array(2, 1), Array(3, 1), Array(4, 1), Array(5, _
        1), Array(6, 1)), TrailingMinusNumbers:=True
```

Given that this is a base model, I didn't get around to using something new that I found out about. By right clicking on the toolbox in the editor, more controls can be found; one of them is the calendar.



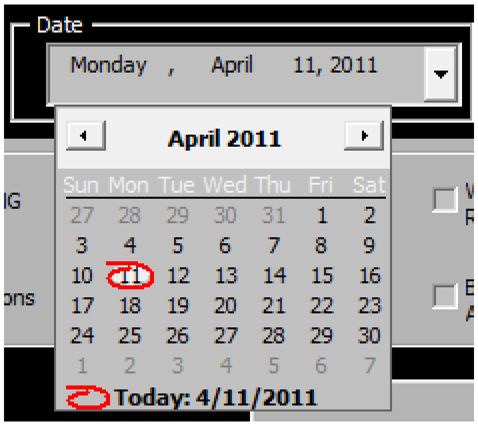
Additional Controls

Available Controls:

- ☐ :-) VideoSoft FlexArray Control
- ☐ :-) VideoSoft FlexString Control
- ☐ Abale Compression Control v5.0
- ☐ Abale Zip Control v5.0
- ☐ ActiveBar Class
- ☐ Adobe PDF Reader
- ☐ AnalogCaptPane Class
- ☐ Apple QuickTime Control 2.0
- ☐ AudioCapture Control
- ☐ Behavior Object
- ☐ Business Contact Manager ShellFrame
- ☐ ButtonBar Class

☐ Selected Items Only

:-) VideoSoft FlexArray Control
Location C:\Windows\system32\VSFLEX3.OCX



This calendar provides an easy solution when the programmer wants to present the option of choosing a date without a lot chance for error on how the date is selected. This base model didn’t utilize the power of this control, but it’s something that could be done to improve upon this base model.

In timing the macro to retrieve the balances from the various bank accounts, it took it about 7 minutes for it to complete the job, that’s seven minutes for what it would probably take me 15 and it’s all done at the click of a button (no passwords to enter into a userform because it pulls it from a spreadsheet). In all, I’m ashamed to admit that this took about 30 pages of code to complete. I know that in part that it grossly exposes my inability to write code in a concise manner but I attribute that to amount of time it took to understand the various websites with which I was interacting. With some more time, it could have been prettier and concise, but the important part is that it gets the job done and I learned a lot.

With the macro that downloads books, I learned a couple things there as well. First, I learned about the ability for VBA to interact with Microsoft in order to create a procedure that would allow the user to select the folder where the files would be stored. To do so, this required the reference of Microsoft Shell Controls and Automation. Second, in the agent parsing through the code to download pdfs, which is something it wasn’t originally designed to do, a sort of backdoor had to be created through the use of sending a cookie to the server in order to download the pdf object from the website. I really appreciate the professor’s help; it taught me a lot.