Ricky Garrard
ISYS 540
April 12, 2010

Final Project – Executive Summary

Executive Summary

For the last two semesters, I have been in charge of the ward directory for my BYU ward. My responsibilities include taking everyone's picture and then creating a picture directory with names and phone numbers. The names are in ABC order by first name. It may take three to four weeks to obtain all the photos, but the Bishop prefers to have an up-to-date copy to help in making church assignments. Since I do not have a program that will auto generate anything for me, I have used PowerPoint to recreate the directory each week by rearranging the pictures as new ones come in. This process can take a few hours each time, especially as the number of photos increases to 200 or more.

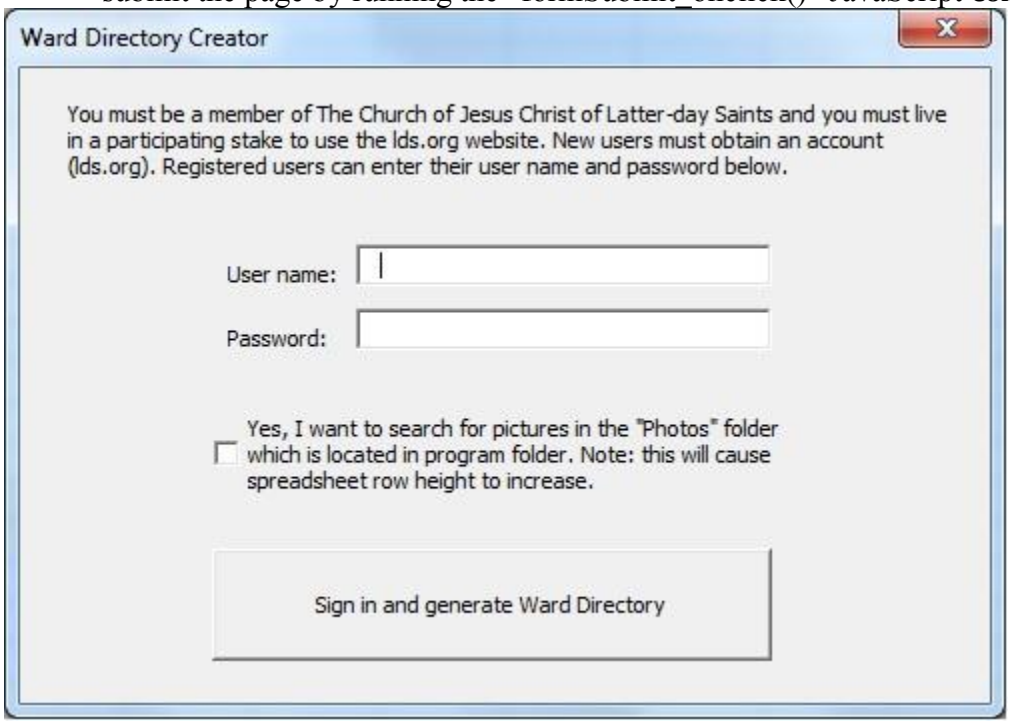
I have created a program that will automatically login to the LDS.org website and scrape the site for ward member data. It will fill in an Excel spreadsheet with each ward member's first name, last name, phone number, and address. Additionally, the program will search the "Photos" folder for ward member pictures based on the filename. If pictures are found, they will be inserted in the appropriate spreadsheet rows. Note: the program has been designed to search for .jpg picture files.

Implementation Documentation

LDS.org Login

When the user runs the macro from the "Login" sheet, he or she will be presented with a login form. The user name and password will be used to login to the lds.org website. The program will then:

- create an Internet Explorer object that will navigate to the lds.org member directory site
- automatically fill in the user name and password
- submit the page by running the "formSubmit_onclick()" JavaScript command (Code #1)



```
'goes directly to ward membership directory site
openPage ("https://secure.lds.org/units/a/directory/1,7946,605-1-7-420344,00.html")

'user name and password
ie.document.loginform.username.Value = globalUName
ie.document.loginform.password.Value = globalPass

'submit form
Call ie.document.parentWindow.execScript("formSubmit_onclick()", "JavaScript")
```

Code #1 – Login code

Scrape Member Data and Store in Array

The program will then use the internet explorer object to obtain the member data from the church records. I chose to scrape the site instead of downloading the CSV file that the site provides so that I could better understand in-class Naxos example. The program obtains the member data by:

- using the "getElementsByTagName" method of the IE document object
- search for "td" tags (table data)
- use "instr" to search for "featurestext" string within the "class" property of the table data
- the program filters out website titles and headings that also contain "featurestext" since that data is not needed
- once correct "td" tags are identified, program uses "innerText" of the "td" tag (contains member data)
- the program then navigates to the next set of last names by looping through and clicking on the appropriate links (A, B, C, D…) and scraping those pages for member data
- program creates an array to store member data as data is scraped from website

```
For Each oneTD In ie.document.body.getElementsByTagName(theTag)

  If InStr(1, oneTD.classname, "featurestext") > 0 Then
  'This potentially contains member data

    If InStr(1, oneTD.innerHTML, "href") > 0 And InStr(1, oneTD.innerHTML, "href=""javascript:emailWindow_onclick") < 1 Then
    'This does not contain member data

  Else
  'This will contain member data

    Debug.Print "membersArray(" & arrayRow & ", " & arrayCol & ") " & Trim(oneTD.innerText)
    membersArray(arrayRow, arrayCol) = Trim(oneTD.innerText)
    arrayCol = arrayCol + 1

      If arrayCol > 3 Then
      'new member
      Debug.Print "-----------------------------"
      arrayCol = 0 'column restart
      arrayRow = arrayRow + 1 'new member row in array

      End If
    End If
  End If
Next
```

Code #2 – obtain innerText from appropriate TD tags

Populate Worksheet with Member Data

After the member data is scraped from the lds.org website and stored in the "membersArray" array, the program populates the "WardMembers" sheet with the data by looping through the array.

Search for Pictures

As the program is populating the worksheet with members' information, it also searches for a matching .JPG picture from the "Photos" folder. This will occur as long as the user has indicated that he or she wants to search for pictures by clicking on the checkbox.

To do this, the program decides on the filename it should be searching for:

```
Function getfileName(namesToSplit As String, lastName As String)
'splits the first and middle name and returns the first name

Dim fName As String

'this will hold the first name in pos 0 and middle name in pos 1
Dim twoNamesArray() As String

twoNamesArray = Split(namesToSplit)

'choose the first name
fName = twoNamesArray(0)

'create the file name that will be searched for
getfileName = fName & " " & lastName & ".jpg"

End Function
```

The getfileName function accepts two variables—the first name and last name data from the church records. The first name may or may not contain the middle name as well, so the program uses the "split" method to add each name to the "twoNamesArray" array. I then use position 0 of the array which will contain the first name only, and then create the file name that should be searched for by joining the actual first and last name. "Ricky Clayton" and "Garrard" should become "Ricky Garrard.jpg".

Here is the code that would search for Ricky Garrard.jpg:

```
Sub TestInsertPictureInRange()

Dim photoPath As String
Dim curPath As String
Dim picRange As String

picRange = "E" & picRow & ":" & "F" & picRow

curPath = ActiveWorkbook.Path

photoPath = curPath & "\Photos\" & fullFileName
    InsertPictureInRange photoPath, _
        range(picRange)

End Sub
```

```
Sub InsertPictureInRange(PictureFileName As String, TargetCells As range)
' inserts a picture and resizes it to fit the TargetCells range
Dim p As Object, t As Double, l As Double, w As Double, h As Double
    If TypeName(ActiveSheet) <> "Worksheet" Then Exit Sub
    If Dir(PictureFileName) = "" Then Exit Sub
    ' import picture
    Set p = Sheets("WardMembers").Pictures.Insert(PictureFileName)
    ' determine positions
    With TargetCells
        t = .Top
        l = .Left
        w = .Offset(0, .Columns.Count).Left - .Left
        h = .Offset(.Rows.Count, 0).Top - .Top
    End With
    ' position picture
    With p
        .Top = t
        .Left = l
        .Width = w
        .Height = h
    End With
    Set p = Nothing
End Sub
```

Once the picture is inserted to the spreadsheet, it has to be repositioned within the E:F range. The picture is not actually contained within the cell.

Insights and Difficulties

I ran into problems every step of the way. It took me a while how to sign in to the lds.org site since I couldn't click an "href" button. I finally realized that I had to use the JavaScript command "formSubmit_onclick()."

Although I obtain the member information in a similar fashion to the Naxos in-class example, I had a difficult time figuring out how I would actually scrape the website since the information is contained within "td" tags and not href links like the Naxos example. At first I was trying to figure out the IE document methods without referring to any online documentation, which was impossible until I found the documentation. The key method I needed was getElementsByTagName.

After finding this method, I still had to figure out a way to pull the member information from the correct “td” tags since other “td” tags contained website titles. This took also took some time to figure out, and my code is included in Code #2.

In order to search for picture filenames I was planning to parse the member information manually until I discovered the Split(string) method. It simplified dividing the first and middle name so I could create a filename that I needed to search for.

Overall, I learned that although html seems pretty cryptic at times, it is not impossible to manipulate and benefit from. It is amazing that we are able to write VBA programs that can access websites and scrape information from them. I would have liked to been able to format the directory so that it looks a little more professional, but I ran out of time trying to figure out how to do everything else.

Here is what the finished product looks like:

	A	B	C	D	E
1	**Last Name**	**First Name**	**Phone Number**	**Address**	**Photo**
2	Abbot	Charles Noe	111-111-1111	1400 N University AveApt 133, Provo, Utah 84604	
3	Anaconda	Sally	222-222-2222	1315 E. 200 No.PROVO, Utah	
4	Bell	Ursula Ariel	333-333-3333	1322 E 100 NProvo, Ut, Utah	
5	Bobby	Ricky Clayton	444-444-4444	1302 E 100 NPROVO, Utah