



# CAR TRADING BUSINESS

Project Write-up

Brandon Leslie

BrandonCLeslie@gmail.com

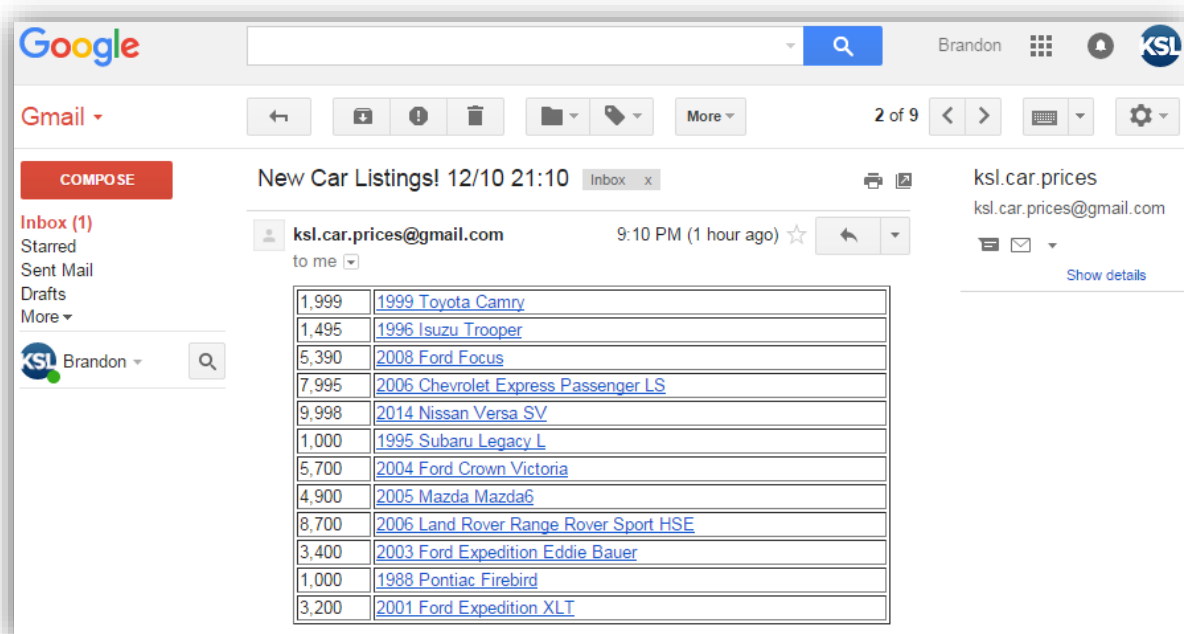
## TABLE OF CONTENTS

Executive Summary.....	2
Implementation Documentation .....	3-7
Discussion of Learning .....	7-11
Assistance .....	12

## EXECUTIVE SUMMARY

Originally, I thought of creating a buying tool for golf clubs that would scrape KSL and promptly send me an email of the latest deals. It was a great idea, until my entrepreneurial roommate came up with one much more lucrative. Instead of searching for golf clubs, we will find the cheap car prices listed on KSL and then compare them to the Kelly Blue Book listings. If the gap is big enough (\$2,000 or more) because some local seller forgot to do their Kelly Blue Book research, then we plan to buy it cheap then turn around and sell it for a big profit. This project was an excellent opportunity to create a tool that would email me directly of the latest deals on KSL.

My entrepreneurial roommate and I hope to pay off the rest of college with our car trading business and graduate BYU debt free!



## IMPLEMENTATION DOCUMENTATION

---

*The following bullet points provide concise documentation for creating my solution.*

- Global variables were necessary to help run information throughout various sub procedures (e.g., between scraping the data and sending the email).

```
Dim a As New agent
Dim htmlString1 As String
Dim finalHtmlString1 As String
Dim List1(199, 2) As String
Dim htmlString2 As String
Dim finalHtmlString2 As String
Dim List2(199, 2) As String
```

- To keep my code organized, I create one subprocedure to select and run the others (Note: this made it easier to run the code every five minutes in emailing new listings).

```
Sub runProgram(control As IRibbonControl)

    kslAll
    kslTacoma

    Application.OnTime Now + TimeValue("00:05:00"), "runProgram"

End Sub
```

- I begin the scraping subprocedures with declared variables. For example, I would need to declare a variable that would hold the string data for the price, name, and link of car found on KSL, so I could then use that same variable to email the data.

```
Sub kslAll()
    Dim price As String
    Dim name As String
    Dim link As String

    Dim iterate As Boolean
    Dim first As Boolean


    Dim i As Integer
```

- So as to prevent avoid any mixing of data from one car to the next, I cleared my Boolean and counters before each car's data was recorded.

```
htmlString1 = ""
iterate = True
first = True

arrayCounter = 0
```

- The KSL URL links were set on the first tab of the excel workbook so I could easy reference that cell to find the webpage. With that URL, I used the agent to open the page link to gather the entire HTML results.

	A	B	C
1			
2	<b>Email Information</b>		
3	From:	<a href="mailto:ksl.car.prices@gmail.com">ksl.car.prices@gmail.com</a>	
4	From Password:		
5	To:	<a href="mailto:scrape.n.sell@gmail.com">scrape.n.sell@gmail.com</a>	
6			
7	<b>Website Links</b>		
8	Newest Cars	<a href="http://www.ksl.com/auto/research/index?keyword=&amp;yearFrom=&amp;yearTo=&amp;mileageFrom=&amp;mileageTo=">http://www.ksl.com/auto/research/index?keyword=&amp;yearFrom=&amp;yearTo=&amp;mileageFrom=&amp;mileageTo=</a>	
9	Newest Tacomas	<a href="http://www.ksl.com/auto/research/index?m_facetClicked=true&amp;m_facetValue=Tacoma&amp;m_facetK=">http://www.ksl.com/auto/research/index?m_facetClicked=true&amp;m_facetValue=Tacoma&amp;m_facetK=</a>	

```
link = Worksheets("Email").Range("C8")
```

- To prevent any errors from stopping my code, I added code to handle errors. The Debug.Print statement was extremely useful while running my code for the first time to notify me of any errors present.

```
On Error Resume Next
a.visible = True
a.openpage link, True
If Err.Number <> 0 Then
    Err.Clear
    Debug.Print "Error opening page #1" & Format(Now(), " mm/dd hh:mm")
End If
```

- After locating the webpage, the next step was to scrape the KSL webpage for new listings. In order to do so, I set the position to 1 and my counter to 0 then moved to specific spots in the HTML document to store the price, name, and link information. The code will stop looping when it has gone through all available listings for that page.

```

a.position = 1
i = 0
Do While a.moveTo("<div class=""srp-listing-body-right"">")
  If Not a.moveTo("href=""") Then Stop
  link = a.getText(""">")
  link = ("http://www.ksl.com/auto/search/index" & link)
  name = a.getText("</a>")
  a.moveTo "$"
  price = a.getText("<")

```

- If there are no new listings on the page, then the Boolean If statement will prevent any emails from being sent. No need for redundancy in my inbox every five minutes.

```

If first = True Then
  first = False
  If List1(i, 2) = "" Then
    first = False
    iterate = True
  Elself List1(i, 2) = link Then
    first = False
    iterate = False
  Exit Do
Else
  first = False
  iterate = True
End If

first = False

End If

```

- If new listings were found, then the price, name, and link information will be temporarily stored into this counter-list of the while loop. Debug.Print was also useful here in seeing what was currently stored in the price, name and link variables.

```

List1(i, 0) = price
List1(i, 1) = name
List1(i, 2) = link

i = i + 1

'Debug.Print price, name, link

Loop

```

- Once that loop is finished, a For loop organizes the data with HTML so it will be neatly shown in a table via email.

```
For i = 0 To 11
    htmlString1 = htmlString1 & "<tr><td>" & List1(i, 0) & "</td><td>" & "<a href=" & List1(i, 2) & """">" &
Next
```

- The message is then stored by the name finalHtmlString1 (finalHtmlString2) and is ready to be sent.

```
finalHtmlString1 = "<table border=""1"" style=""width:100%"">" & htmlString1 & "</table>"
```

- If there is new data to send, then the iterate Boolean variable will be true and run the sub procedure, sendMail.

```
If iterate = True Then
    sendMail
End If
```

- The sendMail subprocedure contains its own local, declared variables.

```
Sub sendMail()
    Dim Mail As New Message
    Dim Config As Configuration
    Set Config = Mail.Configuration
    Dim i As Integer
    Dim Rows As Integer
```

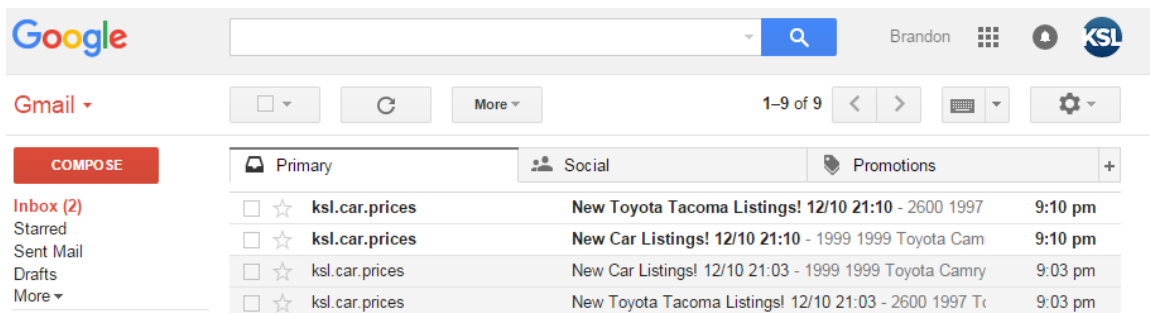
- Configuration settings were then stated, including username and password of the email account ([ksl.car.prices@gmail.com](mailto:ksl.car.prices@gmail.com)) that would be sending these emails.

```
Config(cdoSendUsingMethod) = cdoSendUsingPort
Config(cdoSMTPServer) = "smtp.gmail.com"
Config(cdoSMTPServerPort) = 465
Config(cdoSMTPAuthenticate) = cdoBasic
Config(cdoSMTPUseSSL) = True
Config(cdoSendUserName) = Worksheets("Email").Range("C3")
Config(cdoSendPassword) = Worksheets("Email").Range("C4")
Config.Fields.Update
```

- From there, I established the recipient ([scrape.n.sell@gmail.com](mailto:scrape.n.sell@gmail.com)), email subject line, and body of the message.

```
Mail.To = Worksheets("Email").Range("C5")
Mail.From = Config(cdoSendUserName)
Mail.Subject = "New Car Listings!" & Format(Now(), " mm/dd hh:mm")
Mail.HTMLBody = finalHtmlString1
'Debug.Print finalHtmlString2
```

- Once the sendMail procedure has ran, I receive the message in my inbox.



- The previous steps covered the subprocedure kslAll, which is a current listing of all newly posted cars. The second subprocedure, kslTacoma, is exactly the same except it scrapes a webpage filtered by Toyota Tacoma. Once the emails for both have been sent, this program will repeat every five minutes if new listings are available.

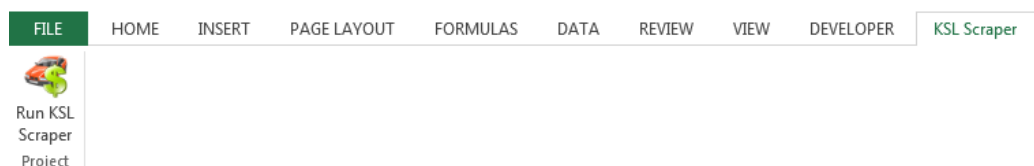
```
Sub runProgram(control As IRibbonControl)

    kslAll
    kslTacoma

    Application.OnTime Now + TimeValue("00:05:00"), "runProgram"

End Sub
```

- To initiate the program, I created a new tab and button from the ribbon customization tool we were given in class.



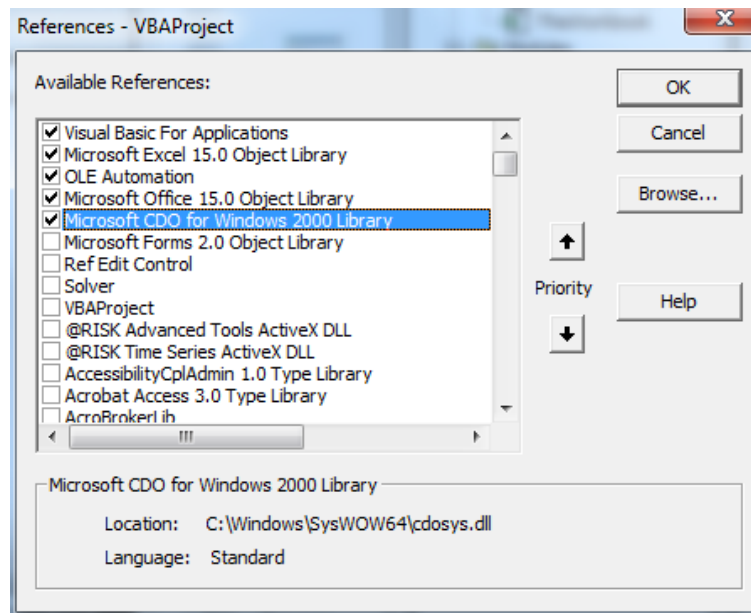


## DISCUSSION OF LEARNING

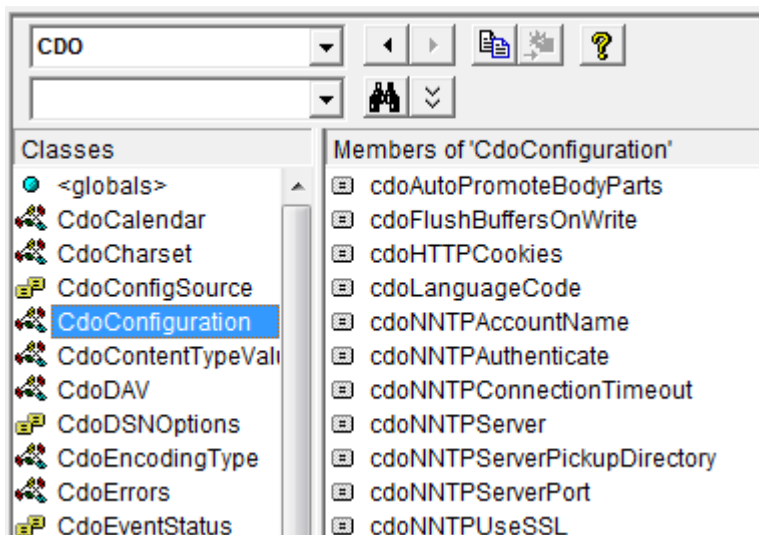
*The following is a list of difficulties and important lessons learned while creating this project:*

### Activate CDO objects

It is important to note that when setting configuration settings such as SMTPServer, the CDO Microsoft objects must be activated. As seen by the image below, they can be activated by going to the References window of the Tools tab in VBA, and selecting “Microsoft CDO for Windows 2000 Library.”

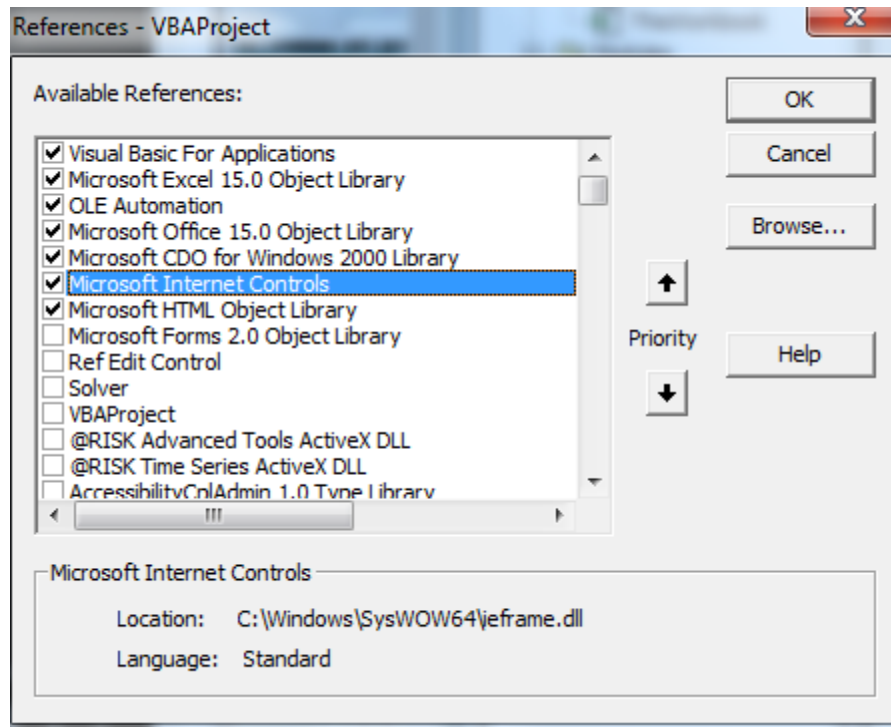


Additionally, the Object Browser button contains a list of all the fields that can be set for email. Below is a screenshot of what that object list looks like.



### Activate objects to pull website data

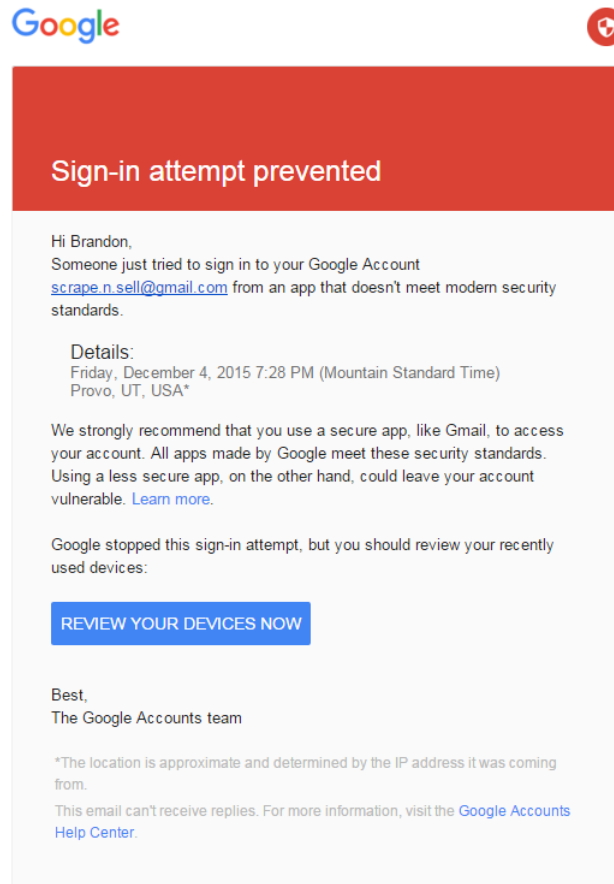
In addition to activating the CDO objects for sending emails, it is also necessary to activate references that allow Excel to pull data from a website. The two available references that will need to be activated are Microsoft Internet Controls and Microsoft HTML Object Library (as shown in the image below).



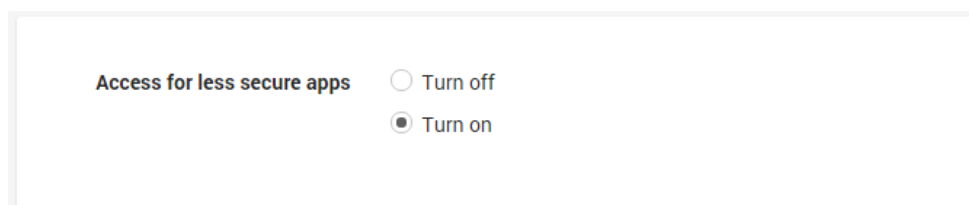
Just as with CDO, MSHTML objects can be looked up in the Object Browser.

## Gmail sign-in attempt prevention

Upon my first attempt of accessing my Gmail account from Excel, I received an email warning from Google saying that “someone just tried to sign in to your Google Account” (see image below).



Apparently, Gmail’s default setting does not allow this type of access to my account from a “less secure app” because “these apps and devices are easier to break into” and blocking them will help keep my account safe. Whereas I took mild comfort in the idea that Google was watching out for me, I was still irritated that my code was not sending me the emails I wanted. To resolve this Gmail issue, I clicked the “Learn more” link (on the image above) which took me to another page titled “Allowing less secure apps to access your account.” From there, I followed the link named “Less secure apps’ section” to a page that allowed me to turn on access for less secure apps, like Excel (see below).



### **Sending email to and from same account**

Everything seemed to be running fine when I first starting running my code, but I was frustrated to errors in sending the email. The code would run but no email would send. To fix this issue, I decided to create another email account, that I named [ksl.car.prices@gmail.com](mailto:ksl.car.prices@gmail.com), then try sending emails from that account. I realized that the code I had written would not send itself emails to the [scrape.n.sell@gmail.com](mailto:scrape.n.sell@gmail.com) address, so I decided to create two accounts. Whereas, I could have re-written the code to send itself emails, this error gave me the idea to create the ksl.car.prices account which looks more professional (as opposed to seeing your own email address in the “From” column of Gmail). Morale of the story, working through this error helped my project look more professional.

## ASSISTANCE

---

Most of the code used for sending emails from Excel using VBA and Gmail came from a YouTube user named DontFretBrett. His tutorials are easy to understand and straight to the point. Specifically, I found the following video to be extremely useful: <https://www.youtube.com/watch?v=cOhupITOrNA>. Additional coding was found by studying the code of past projects submitted onto our class project blog.

I am very grateful for Dr. Allen's help in scraping the data. I attempted to use code found online, but immediately became lost and confused with the scraping processes of other users. Dr. Allen showed me a quick way to access the KSL webpage and scrape the price, name, and link information that I needed. Additionally, the RibbonWizard and Web Agent tools from our course work proved extremely helpful.