A.J. Bronson
IS 520

Executive Summary

I chose to do my project on KSL car searching. When looking for a new car, the process can become mundane and take a lot of time. I decided that the scope of this project would be to write a macro that would scrape KSL's website looking for a specific make and model of a car. The query would produce results and show them to the end user on an excel worksheet, as well as sending an email and text to the user.

Implementation Documentation

Externals

In order to execute this project, I needed a few external modules. I used the agent class, and the sendGMail and text functions written by Dr. Gove Allen. I also used the email validation function found at:

http://www.vbaexpress.com/kb/getarticle.php?kb_id=281

User Form Search Parameters

I began by creating a userform (see Exhibit A) to accept basic and advanced parameters. The basic parameters include details such as: make, model, price range, mileage range, new or used, etc. The advanced options include details such as

whether the vehicle has a clean or rebuilt title, how long the ad has been posted, and the color and condition of the car.

I thought about the correct way to receive all of the stated parameters, and decided that I would like to avoid user error wherever possible. The only field in which the user can enter text on the entire form (besides the notification center) is the "Keyword" textbox. All of the other parameters come from drop down menus (combo boxes) where the user has a hard coded list of options. I decided that I didn't want to take complete control away from the end user, so I populated these combo boxes from the data contained in the worksheets named "MakeModel" and "Options".

The "MakeModel" worksheet lists all of the different car manufacturers across the top row. Under each manufacturer is a list of vehicles that they produce. Programming the form this way allows the end user to add new vehicles and manufacturers to the list, or to remove options that they know that they will never use.

The "Options" worksheet contains all of the other combo box data. The form relies on the options in *this* worksheet to be in their exact original location. However, the end user may still add to these options by typing additional items to the end of each list, or by modifying the year and price section. I wouldn't recommend making changes to either the "MakeModel" or "Options" worksheets unless completely necessary, as the query to KSL requires these fields to be exact.

User Form Notification Center

In the form, there is a frame titled "Notification Options". After a query has been run, all results are printed to the workbook in Sheet1. However, if the user would like additional notification, they can either select the "Email", or "Text" checkboxes, or both. If the "Text" checkbox is selected, I included validation to make sure a 10-digit phone number has been entered, and a mobile carrier has been selected. If the "Email" checkbox is selected, I included validation to ensure a correct email was entered. In this project, a correct email means there is some text, the @ symbol, some text, a period, and lastly some additional text. This email validation was pulled from the Internet. Both the text and email validations will inform the user if they have made a mistake.

If the "Text" checkbox has been selected, and a valid cell phone number and carrier were provided, the user will receive a text message after the query has finished informing them that the query executed successfully, and telling them the number of results and average vehicle price for the resulting list.

If the "Email" checkbox has been selected, the user will receive an email when the query has finished. The email will include the title, mileage, price, and a hyperlink to each vehicle result from the query. If the user had entered a number into the "Advanced Notifications" section, the vehicle results sent to the user's email are filtered by those inputs. The available filters in "Advanced Notifications" are: less than an entered dollar amount, and less than an entered mileage.

The most important part of this section is the checkbox titled "Send Separate Email with Good Deals". When clicked, it forces the "Email" Checkbox to be checked

and an email to be entered. After the average price and mileage have been

determined from the query results, the source code will iterate through each vehicle

and flag it as a good deal if its price and mileage are less than the average price and

mileage of all vehicles within the results. All good deal vehicles are then highlighted

in blue on the excel worksheet. In addition, if the title "Send Separate Email with

Good Deals" checkbox is selected, the user will receive an additional email with just

the good deal vehicles.

The Query

When the user presses "Search KSL" from the KSL cars ribbon, the run_ksl

sub procedure is called, opening the user form. After pressing "Search" on the form,

all data validation occurs. After data validation, the run_ksl sub procedure

continues, opening the agent class module. The agent class then opens a KSL static

webpage based on the parameters entered into the form. The agent will figure out

how many pages there are in the query, then grab each vehicle from each page,

loading several pieces of information on that vehicle into a two-dimensional array.

After executing the query, the results are printed and formatted onto the

Sheet1 worksheet (see Exhibit B). If the sub procedure needs to send emails and/or

a text, it does so now. The email message content comes from iterating through the

two-dimensional array containing vehicle information.

To the right of the printed results on Sheet1, the sub procedure will print a

few pieces of information. It enumerates the total results, max, average, and min for

price and mileage, and the success or failure of emails if applicable.

Other

In order to begin the search, I created a custom ribbon tab titled "KSL Cars". Within this tab, there are two buttons. "Clear Previous Results" clears everything out of Sheet1. "Search KSL" begins the sub procedure run_ksl, and shows the user form.

Conceptual Difficulties

One of the things that I would have liked to include in this project was a progress indicator as soon as a query begins. Occasionally when the query parameters result in several hundred vehicle matches, the sub procedure must run for several minutes. I searched the Internet on how to implement this, but everything that I read seemed a little over my head. One of the things that I learned through this project was that doing a comparison of an integer stored in a string variable and an integer stored in an integer variable won't always throw an error, but it will not evaluate the comparison correctly.

Assistance (no assistance)

I received no assistance from any students during this project. The only assistance that I received was enumerated in the section titled "Externals".
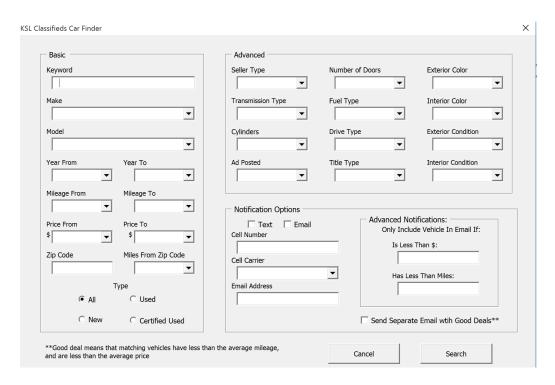
Appendix

Exhibit A

Exhibit B