

Andrew Brasher

# Craigslist Apartment Search

VBA Final Project

## Executive Summary

Every year around March my wife reminds me that our housing contract is going to expire and that we need to start looking for housing. So I open my browser and go to Craigslist, then search for the same criteria again. After I get the results, I open a bunch of tabs and look at each ad separately. Then I try to remember to save each potential website in my bookmarks or I transfer the information to a Google doc to share with my wife.

I thought that this long process could be easier if I could just run a search, import the information in a way that makes sense, and save the document. For my VBA Final Project I made a program that will allow a user to search Craigslist. The program will import the information and any available pictures to the website then the user can save the worksheet.

## Implementation Documentation

In order to implement this program there was admittedly a lot of trial and error. I estimate that I ended up rewriting the code for this project three or four different times. This was necessary because of several “exceptions” that Craigslist allows through their site. These different exceptions will be discussed later. However, at this point I will discuss each step I used in the Craigslist scraper.

When the user first opens up their excel program they are shown a simple button. By pressing the button all the magic of the program begins. The first important step of the program is part of the user form initialization. The initialization performs three specific tasks: (1) It clears any previously saved information from the spreadsheet. (2) It deletes any pictures which may have been saved on the spreadsheet, see in Figure 1. And (3) it populates the bedroom combo box with the same option from Craigslist.


Ad #2		
Title:	\$315 / 3br - HUGE Master Bedroom	
Date:	2011-04-06, 1:38PM MDT	
Email:	keaira6@hotmail.com	
Location:	Provo/Urem Border	
Website:	<a href="http://provo.craigslist.org/apal/2309272851.html">http://provo.craigslist.org/apal/2309272851.html</a>	
Description:	Amazing Master Bedroom up for rent! Great location, adorable house. There is private covered parking, extra storage space, master has own bathroom/shower, two huge closets, house includes full kitchen, washer/dryer, and is there are two other rooms currently rented. Roommates are really fun, great girls, I am only leaving because I have to. This is a great deal and the contract goes through August with the ability to resign. BYU standards please. Contact Kara at 503-702-4688 for further details.	
*Double click on the description to view its entirety		

Figure 1



Figure 2

This then allows the user form, seen in Figure 2, to be displayed. The user then would enter the proper information in the different text fields. Then they press the “Search” button. Since Craigslist does not require any authentication when utilizing its search options there is no need to populate the search terms in the Craigslist website, but instead we just populate the search terms directly into the URL, as seen in Figure 3.

```

If Not bedroomNum = "0" Then
    site = "http://provo.craigslist.org/search/apa?query=" & searchCriteria & _
        "&srchType=A&minAsk=" & minRent & "&maxAsk=" & maxRent & "&bedrooms=" & bed
    agent99.openpage site
Else
    site = "http://provo.craigslist.org/search/apa?query=" & searchCriteria & _
        "&srchType=A&minAsk=" & minRent & "&maxAsk=" & maxRent & "&bedrooms=" &
    agent99.openpage site
End If

```

Figure 3

The next step is to pull the necessary information from the first ten search results (if there are less than 10 results the program will quit after displaying the final search result.) The first part of this step is to prepare a location in order to save the images that will be potentially pulled from each apartment ad. As seen in Figure 4, this is done with the “MkDir” command. This command is commonly used through the command prompt to create folders, but can also be used through VBA. In this instance I hope that by giving the folder a more unique name then “images” I be able to create the folder without any error.

```

'create folder for images
MkDir "C:\cListImages521409731\"

```

Figure 4

Now we are ready to gather the information from the website.

Originally when I wrote this program I would import each ad to excel, then find and pull each string of information

I needed. However, I encountered several exceptions to the “rule” as I went through the process of finding where information was stored on the imported excel page. This will be address later. I then found in order to find consistent results and avoid the hassle of error handling; I needed to pull the information directly from the website page source.

The agent variable, with the class module, provides four specific functions which I used the most. They are called:

1. agent.position
2. agent.moveTo()
3. agent.moveBackTo()
4. agent.getText()

To explain how these methods work one must understand that when the agent variable goes to a website and views the page source it actually reads the entire page as one long string. By entering in the option agent.position = 0 this places the agent at the zero position on the string. Combined with using the agent.moveTo(“”) you can move along the string, or through the HTML page source, to any tag that contains the needed data.

This is first used to gather the website link for the first ad in the search results. An example of a search result is in Figure 5. In the page source each search result is prefaced with a "<P>" tag. This is unique throughout the entire page for just the results. Each result line

contains three pieces of helpful information at this point. (1) We can gather the ad website URL and verify if there is another search result in the list. (2) We can get the title for the ad. And (3) we can get the location (if provided) for the apartment.

```
Apr 6 - $315 / 3br - HUGE Master Bedroom - (Provo/Orem Border) pic
Apr 5 - $300 / 1br - MEN'S PRIVATE APT NEXT TO UVU - (Summerwood, Orem, UT)
Apr 4 - $300 / 1br - 4 month summer contract (may-august) awesome apartment! - (543 north university ave. #206)
Apr 4 - $330 3 Private Mens @ The Belmont - (BYU) img
Apr 4 - $330 / 3br - Belmont 3 Private Mens - (BYU) img
Apr 4 - $335 1 shrd FWSS Mens - (BYU) img
Apr 4 - $350 / 4br - Summerlynn Condos - (BYU) img
Apr 4 - $310 / 3br - 4 shrd 1 private - (BYU) img
```

Figure 5

```
<P>Apr 6 - <A href="http://provo.craigslist.org/apa/2309272851.html">$315 / 3br
- HUGE Master Bedroom</A> - <FONT size=-1>(Provo/Orem Border)</FONT> <SPAN
class=p>pic</SPAN></P>
<P>Apr 5 - <A href="http://provo.craigslist.org/apa/2307996960.html">$300 / 1br
- MEN'S PRIVATE APT NEXT TO UVU</A> - <FONT size=-1>(Summerwood, Orem, UT)
</FONT></P>
<P>Apr 4 - <A href="http://provo.craigslist.org/apa/2305510356.html">$300 / 1br
- 4 month summer contract (may-august) awesome apartment! </A>- <FONT size=-1>
(543 north university ave. #206)</FONT></P>
<P>Apr 4 - <A href="http://provo.craigslist.org/apa/2305305712.html">$330 3
Private Mens @ The Belmont</A> - <FONT size=-1>(BYU)</FONT> <SPAN class=p>img
</SPAN></P>
<P>Apr 4 - <A href="http://provo.craigslist.org/apa/2305303077.html">$330 / 3br
- Belmont 3 Private Mens</A> - <FONT size=-1>(BYU)</FONT> <SPAN class=p>img
</SPAN></P>
```

Figure 7

```
'get location
agent99.moveTo ("</P>")
If agent99.moveBackTo("<FONT size=-1>") = True Then
    Dim comparead As String
    agent99.moveBackTo ("<A href=" & Chr(34))
    comparead = agent99.getText(">")
    comparead = Replace(comparead, Chr(34), "")
    If comparead = adwebsite Then
        agent99.moveTo ("<FONT size=-1>")
        location = agent99.getText("</FONT>")
        location = Replace(location, "(", "")
        location = Replace(location, ")", "")
        'Debug.Print location
    Else
        Do While Not comparead = adwebsite
            agent99.moveTo ("<A href=" & Chr(34))
            comparead = agent99.getText(">")
            comparead = Replace(comparead, Chr(34), "")
        Loop
    End If
End If
```

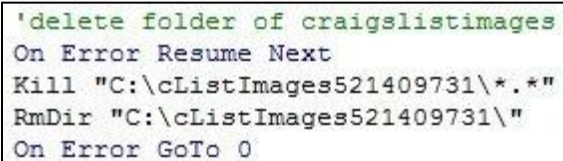
Figure 6

A helpful way to see how this works is to look at how to get the location information. In Figure 6 we can see the page source and in Figure 7 we will look at the code to get the location for the correct ad. At this point our agent99 variable is position after the closing "</A>" tag (circled in red on Figure 6). We then move the agent99 variable to the end of the ad line by going to the closing "</P>" tag. We then backtrack to "<FONT size=-1>" since this is where the location is displayed. However, some websites do not display their location so we run the risk of back tracking to the

previous ad's location. To avoid this we have to verify if the website address where we are currently positioned is the same as the website that we are trying to get the address for. So once again we will back track to the "<A href=>" tag located at the beginning of the search result line. We will then use agent99.getText(">") to get the URL for the ad website. By comparing this ad website to the URL we gathered prior to this moment we can verify if they are the same. If it is not the same, we will enter the ad website and get the other information we will need. If the ads are the same website then we will move forward to the location tag again and pull the information between the "FONT" tags. At this point we cannot gather any more information from the search page. So we use the agent to now open the ad website we had gathered previously.

Now that we are actually in the ad website we can gather the next set of necessary information for the user. The next set of data we need to gather are as follows: (1) The postdate of the ad. (2) The email address for the ad. (3) The description for the ad. And (4) any available images. As before, I used the position and moveTo option to go through the HTML page source to gather the required information. Each set of data gathered will then be populated into the excel results spreadsheet. Lastly, by using the agent.moveTo option we can verify if there are any images on the website. If there are images then each image URL is copied and then used to save the image to the previously created Craigslist image folder. While each image is saved to the computer it is then copied to the excel worksheet to be viewed by the user. For later assistance, each image is also given the name of a shape item. This will then allow the program to delete the images when a new search is performed since images are not deleted by simply clearing a spreadsheet. Once this is all completed, the program returns to the original search results website to repeat for the next search result.

Lastly, once all the search results are populated into the excel sheet we need to delete the images and folder from the user's computer. Figure 8 demonstrates this visually. The "Kill" command is used to delete each file in the folder, which is necessary since the next command, "Rmdir", cannot delete any folder with any data stored in it. By deleting all the files within the Craigslist images folder the "Rmdir" command can then remove the directory path resulting in the folder deletion.



```
'delete folder of craigslistimages  
On Error Resume Next  
Kill "C:\cListImages521409731\*.*)" "  
Rmdir "C:\cListImages521409731\" "  
On Error GoTo 0
```

Figure 8

## Discussion of Learning

In this project there are several key areas where I developed a new understanding of VBA limitations and abilities. In the following I will discuss several of the problems encountered during this project and some of the solutions that I attempted.

The biggest problem I encountered through the project was pulling the information from the website. At first I imported the search results into a worksheet and then also imported each ad website to another excel spreadsheet. This made viewing data really simple, but entailed other problems. The first problem was caused by inconsistencies of how Craigslist displayed apartment information. This is demonstrated several ways. For example, some apartment websites use "Location" to title the location

while some use “Address”. With using the normal find function in VBA the program would often crash or break since “Location” wasn’t in the document. I had tried to use error handling techniques but they often didn’t work correctly, although I admit that it could have been my fault for not using the error handling techniques correctly. This was solved, as explained previously, by using the agent.moveTo methods in the website page source.

Another example importing the page caused to my program was because of Property Solutions Management. Property Solutions has found a way to enter HTML Styling code directly into the description area. It then displays on Craigslist as an image (see Figure 9 for an example), even though there is no image, and moves information to different columns when imported. I was able to address this through the website page source and returning to the user information stating that the description cannot be displayed and will need to be viewed on the website directly.



Figure 9

The next problem I encountered was copying images from the apartment advertisement to the excel sheet since they are not imported when you import a page. This was solved again through using the agent variable instead of the page import. By moving to the correct portion of the HTML page source I was able to search for the image and more easily determine if there actually was any image from the HTML tags. It is also important to mention that as I was finding the solution to this problem, I also encountered the other problem with creating a folder to save the pictures in. I did not want to create a folder and leave a bunch of random images saved on the user’s computer. After some research I found that by using the “kill” file command and “Rmdir” command I could delete the images and the folder so that it could recreated when the user performs another search.

Another problem I encountered was when a user wanted to perform a second search. This is common since new posts appear on Craigslist every day. I created an empty copy of the results worksheet, called “beginner”. This means that every time the “Search Craigslist” button is pressed the code will copy the empty worksheet and paste it over the results worksheet. This solution was simple enough, but did not resolve the problem with deleting any images from previous searches. I discovered that the images are

copied to excel as shapes. So when the user presses “Search Craigslist” the program will also go through and delete every shape on the worksheet in order to remove all the photos.

The last problem I encountered through this process was caused by websites not loading fast enough. Although the VBA code prompts for the program to wait till the website is loaded it doesn’t always “wait”. This was made apparent when one of the apartment posts was imported as a duplicate to the search results import. By not importing the pages and by using just the agent to move position to different tags it allowed the site to load completely prior to any actions being performed.

At this moment, there is no problem in the code that I could not figure out how to solve. However, some additional features which would be nice to implement outside of the scope of this project would be to have the program email the apartment owner a stock request for more information about items like washer/dryer, utilities, and the BYU ward. Also, I could add in the option to search for posts with images only, but because of the fake image created by Property Solutions I don’t believe it would be helpful to add in such a search feature.

## **Conclusion**

This project contains all the functionality defined in the original scope of the project. It was a surprisingly complex solution, but it forced me to learn how to use the more robust tools that come with the agent class. I mostly enjoyed seeing the successful results displayed consistently while handling the exceptions.