AutoBib 1.0 Readme

I signed on for a job in the Eastern Michigan University library downstairs in cataloging awhile back, it was a position where a student worker had to go through the manual entry of a series of 7 digit numbers, thousands of times into a bibliography database and check to make sure that it was both on file(or not) and if the bibliography database linked to the right place. after actually doing this the way it had been done thousands of times before, I finally decided this would be a perfect opportunity to utilize some of the things I had been learning in my Computer science classes, so I took it on myself to both learn how write a task automation program and actually do it.

The first step was to find a software that would allow task automation through programmed macros, the software I found which fit best was AutoIT, which is a software package that you will need if you are trying to get this software to work at any other location than Eastern Michigan University’s Cataloging department.

The task is for student workers to take a list of bibliography numbers, enter them in manually into the voyager cataloging system, check to see if it

found the bibliography number in our database, if we did have it, we were to then check the hyperlinks to ensure that the target document was the

document we were looking for. i then had to enter in a "delimiter x" followed by the tag "zxcv" to label the bibliography document as "correct".

then we had to close out of the stuf and redo the same task hundreds of times. below is the breakdown of what exactly this program does on a step by step basis.

as well as a user guide, so you know the notable things and how to use this program.

System Requirements: a PC (running windows 7) that is moderately fast that can deal with having multiple programs open at once, a high speed internet connection.

Overview (Simplified):

This program automates and synchronizes the use of textpad, internet explorer, Voyager, Adobe Reader, Microsoft word, and microsoft excel

in order to check a list of bibliographies for accuracy, and then populate a microsoft excel document with the findings as you go.

The software search algorythm manages to be very accurate, however there are a number of instances where either the time it would take to develop a

solution to a particular event would have taken too long to warrant putting the required amount of time in the fix the issue, or there are instances where the

methods used in the algorythm won't work due to inherent problems with the individual programs that are being run in tandem with this macro software.

Current issues that can result in a negative result even though it may actually be a positive result.

-mispelling/not punctuated correctly in either the bibliography in voyager or in the document itself (the finding algorythm utiizes a ctrl+f feature of adobe reader

as well as internet explorer, so minor typos can provide a false negative result), this is actually a benefit because it allows you to go back and make corrections to

bibliographies if they have a descrepancy.

-if the hyperlink links to a file structure or directory instead of linking directly to a text file or pdf (the time it would take to write a program to test for and

deal with this kind of scenario didn't warrant the time due to how infrequently this happens)

-if the title has any sort of quotes in the name (quotes are used as booleans in the Ctrl+F feature, which messes up the algorithm, so if the program comes across

this issue with the title name, then it will write "QERROR" in all of the fields in the excel document in that row.

Notes:

This Version Does not check for Hyperlink validity for records that were previously checked and tagged properly, that is handled by another member of the department accordingly.

Make sure scroll lock is off, and caps lock is off

Line 528 in program is set to sleep for 25000 milliseconds in order to allot enough time for downloading the pdf file/ opening it up in adobe reader, the computer this was all programmed on is pretty slow and the internet is a little flaky sometimes in regards to it’s speed so this was done in order to accommodate for that. For people with faster computers, you could fine tune this number to better work for you (because it seems like a slow wait in it’s current state) but for reliablity’s sake, leaving it at a decent length wait time will help accuracy, because then you can be sure the program is working correctly when you walk away from it.

The nice thing about using Microsoft word and Microsoft excel (2010) is that it is set to autosave, so even if the power does go out, you can open the document that was being worked on and get your data back.

How to Use The Program:

Step 1. Open the AutoBib program

Step 2. Read Through the whole readme to understand what this program does and how to use it

step 3. Watch the associated help videos if you’ve never used this program before at “” (if you are running this as a student assistant in the EMU Cataloging department, then it has already been Pre-Configured so you can skip those steps)

If you are a new user or a user who is running this program on a different system, you will need to calibrate the X and Y coordinates (instructions in next step) as well as change the website address internet explorer is linking to. You will also need to make sure file associations and defaults are set up properly (which will also be covered in step 4).

step 4. Make sure everything is properly configured by following the directions in the videos on youtube

step 5. Enter in your Voyager Username and Password in the AutoBib program in the appropriate boxes then left click on the “sign in” button

step 6. Click the “Open Files” button, if everything is properly configured, it should open all of the windows you need and log you in to voyager

step 7. Do a bib search in voyager and hit the save button, when/if prompted with a dialog box that asks you if you want to change your Import/Replace profile, click no. then exit out of the voyager document (not the voyager program) by clicking on the smaller grey “x” underneath the window controls in the upper right hand corner of your screen.

Step 8. Now reselect the open autobib program window and hit start

Step 9. Congratulations! The program should be working A-OK

Step 10. You will want to check on the program periodically, to make sure it didn’t get hung up, This is a tool programmed with the sole purpose of allowing me to streamline the process and drastically reduce the amount of involvement this process takes, so one can accomplish other tasks while simultaneously getting link checking done. This program is not exhaustive as there are an incredible number of discrepancies or issues that can happen along the way, But as it stands now, it can properly deal with around 90% of the cases, leaving only 10% of your original list (which is quite a timesaver if you have thousands to do) for you to go back and recheck for any issues (you only really need to check the ones that came back with an “N” in the proper link categories).

Step 11. If the program does get hung up, hit the “Pause/Break” key to pause the program, stop the program from running by “”, I would then manually enter in the issue in the excel spreadsheet so I could further investigate later, then save a numbered iteration of the document in the proper place, rename the documents back to what they were named before, delete the excel spreadsheet’s data and scroll back to the top of the document, close out of any voyager or pdf documents you have open (leave the programs themselves open), and then you should be good to hit start again (there is a video of this process online).

Step 12. When the process is complete, you can copy and paste all of the excel documents together, and then you should have an ordered list off all of the links that were checked by the program. Like I said before, now is when you would go through the document again to do your own searches for anything that “was found” but came up with a “N” for the link (which means the title for the 245 field wasn’t found in the target document). This can be due to any number of reasons, the instances I’ve found that this happens with are listed below:

1. The PDF could be a protected document, in which case the copy function won’t work on the pdf and it needs to be checked manually because copying what is found is an important part of the algorithm.
2. The pdf could be just a series of scanned images in which case ctrl+f wont function properly as there actually isn’t any “computer readable text” to be searched.
3. Another option may be the document took far too long to load off of the webpage, it didn’t load in time.
4. The hyperlink actually sent you to a website that does search functions, or is a file directory, which is out of the scope of this program because it would take too long to program a way to deal with those issues.
5. And finally there is a chance that there is actually a typo, or punctuation/ words used in the Bibliography record that doesn’t match the target document.
6. There were quotes in the title, because that messes up the ctrl F function (those cells will be marked with the label “Q-Error” which stands for “quotes error”)
7. In our library when a librarian goes to check the “delimiter x zxcv” tag validity they put a date after the tag, this will also report a negative.

Step 13.

Current Bugs I’m working on:

It would appear that after a Q error happens, the program will start kicking back false negatives

The program will get hung up if there is no 856 field to find

Step By Step Rundown of what this program does:

1. Creates a function to pause the program while it’s running so if there is a problem you can deal with it
2. Has a section devoted to X and Y coordinates to properly configure mouse movement so users can change the settings in order for this program to work on other machines with other configurations .
3. Declaring and Initializing Variables
4. Creating and linking the files used in the GUI(Graphical User Interface)
5. Button to Display this readme document
6. Button to display setup information
7. Fields for user to enter voyager login info and submit info to variables for use by the “Open files” button
8. The “open files” button opens all necessary programs for the program to work and maximizes all of the windows
9. The “start” starts the main program
10. Resets necessary variables and arrays for next iterations of loop
11. Opens the bib search box in voyager
12. Copies the bib number from word
13. Ends main function if the end of the word document has been reached
14. Pastes the Bibliography ID in Excel
15. Pastes the Bibliography in voyager search box and searches for it
16. If the Bib# isn’t found by voyager, it makes note of it in excel, moves onto next row and goes to next loop iteration
17. If the Bib# is found it moves onto next steps
18. Loop Iterates through first column in voyager bib Document to find the number of Hyperlinks by counting the number of instances of “856” rows (by copying cell data to variables and then comparing the findings to a pre-set variable), saves number to drive # of links to check and record information for, for later.
19. Loop iterates through first column again to find the first 856 row, then enters keyboard commands accordingly to copy the data int the hyperlink field associated with the 856 row, copies what it finds to a variable, checks if delimiter tag is located inside of string variable, if it is, it stores result in an array and does the same thing with any other hyperlinks (if applicable, based off of previous 856 number) it makes note of it’s findings in excel, if it did find the delimiter tag, it closes out of the voyager document and goes on to next bib#, otherwise it makes a note in excel that it wasn’t already checked and continues on.
20. Next voyager is reactivated, keyboard commands send the mouse back to the correct column
21. It then iterates upward, copying cell data until it finds 245 (the title field), moves over to the title field cell to the right, then selects and copies the whole thing
22. Everything except what is found between the first and the second delimiter is trimmed off and the result is stored in a variable (this is done a second time too with a different trim length in order to deal with the occasional improperly formatted 245 data)
23. \*\*\*Then the remaining string is checked for quotes, it quotes are found Q-error is written in the remaining excel columns and the program moves onto the next bib# (because quotes are used as booleans by Ctrl+F I made the program write that out so it can be manually checked later). Otherwise if there are no quotes the program opens the hyperlink window and clicks on the first link (additional links will be clicked after the text/pdf process is complete, necessary # of links to be checked is based off of # of 856’s found, and is driven by a loop that increments the y coordinate value of the mouse movement based off of the number of 856’s found)
24. If the file is a pdf document, it will open in adobe reader, if it is a text document it will open in internet explorer (set as default browser because it will force adobe reader to handle the pdf documents, instead of firefox which will use it’s own pdf reader which doesn’t work nearly as well as adobe reader)
25. the program then waits to a lot enough time for the pdf to download/ the website txt document to load, then hits ctrl+f to open the “Find” function in the document.
26. The trimmed title from the 245 column is entered into the search field and the program hits enter so it will search the document
27. If it finds nothing, it will leave the value of “default” in the proper array and closes out of any “word” not found prompts in adobe reader, Otherwise if it does find something, the keyboard hits escape, the word is highlighted and whatever is found id copied to “found” variable
28. Both the found variable as well as the previous trimmed variables are entered into textpad in order to remove any program specific text formatting (so that a proper comparison can be made) and recopied to their original variable
29. Then each of the saved variables are stripped of any spaces, punctuation, weird characters, etc. in order to allow the program to properly check if they are equivalent.
30. then both trimmed variables are compared to the found variable, and if either of the trimmed variable compared to the found variable is a match, it will make note of a correct match in another array.
31. This process is repeated for any additional links
32. Wtf line 553 & 554
33. Loop Iterates down the list to find 856 in voyager again, when it is found keyboard commands are automatically entered to select the proper cell associated with it, and if the link was marked as correct in the array, it is marked with the proper delimiter tag, if wasn’t marked as correct, it doesn’t enter in any delimiter
34. Repeats as necessary for other links too
35. Mouse moves up and saves the record in voyager
36. Closes out of current bib document(not voyager, just the document)
37. Makes excel window active again, populates the rest of the row with it’s various findings
38. Theres a loop to make sure the excel document is properly set up to do it’s next iteration by adjusting the number of keyboard strokes it will take to get the next intended excel cell selected based on the # of 856 rows that were checked
39. The program closes out of any open documents in any of the original programs in needed to run (but not the programs themselves) in order to prepare it for the next loop iteration
40. The main function stops
41. The x button on the top right corner of the program closes the program

Known Bugs (Currently working on):

1. After a Q-Error, the program will log false negatives in excel