Donovan Hubbard – Searching for MMA fighters

Executive Summary

Mixed Martial Arts is one of the fastest growing sports in the world. As with most professional sports, many people want to compare the athletes' statistics against each other. While it is not difficult to view one fighter's statistics online, it can be difficult to compare hundreds of fighters' statistics simultaneously, especially if you want to try and search for all the fighters that meet certain qualification.

In order to solve this problem, I wrote an excel vba project that downloads fighters statistics off of the website fightmetric.com and allows users to perform more comprehensive searches.

First, the worksheet downloads 20 different statistics on each fighter on the website. Then it gives users an advanced search form. It allows them to search numeric values using the logical comparators of less than, equal to or more than. It supports up to five different simultaneous search filters.

Once a user has found the records that they are looking for, they can export the data in two different ways. One way is to save the results into a new worksheet. The second way is to attach the results to an email and send them using Yahoo or Gmail.

This project interacts with the user through four different user forms that have robust error handling.

Introduction

Mixed martial arts is one of the fastest growing sports in the world. Simply put, mixed martial arts (MMA) is a sport that combines boxing, wrestling, jiu-jitsu, karate and any other combat sport. Fighters face off in a ring until someone is knocked out, someone taps out, or the time limit expires.

There are multiple MMA franchises that have their own fighters and titles, but the number one franchise in the world is the Ultimate Fighter Championship (UFC). Many fans have an interest in tracking how well fighters are preforming. Some people just want an objective way to compare fighters, and others have fantasy teams that they are trying to maximize. In order to view the detailed fighter statistics, users can go to the UFC's official database hosted by fightmetric.com.

It can be difficult and tedious to find all of the fighters that you are looking for, especially if you want to compare current fighters to retired fighters that haven't fought in years. My vba project for the semester was to write an excel sheet that will allow you to search and compare fighters.

Overview

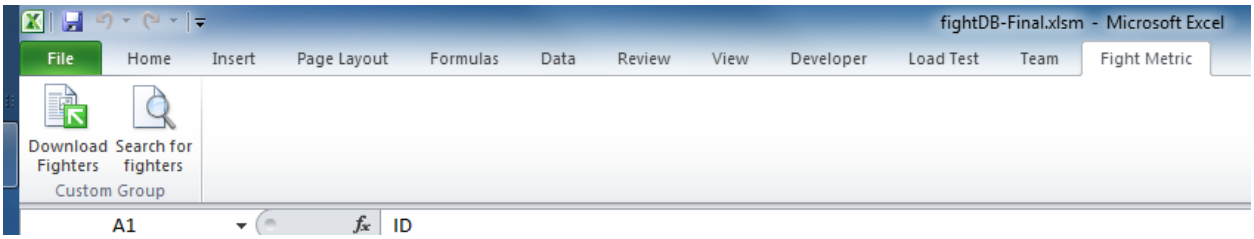
The excel file fightDB.xlsm is comprised of several parts.

1. A subroutine that downloads the data from fightmetric.com.
2. A user form that searches the fighter statistics in the workbook.
3. A function that allows users to export the search results to another document or email.

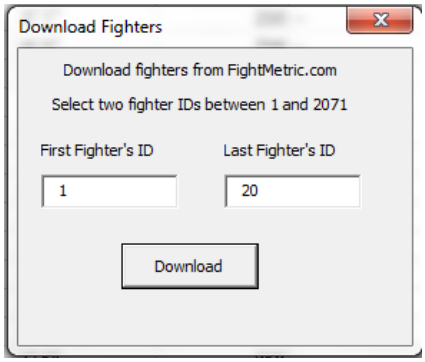
Download Fighter Data

To start your analysis of fighters, you need to download the data. Fightmetric.com has detailed data on every fighter that has appeared in the UFC since the year 2000 (2071 different fighters are in the database), and has assigned each fighter a unique ID. This excel project uses those unique IDs as primary keys for downloading the fighters and sorting among them.

To start the download, a user will need to use the custom ribbon and buttons at the top of the excel sheet, as demonstrated by the following screenshot.

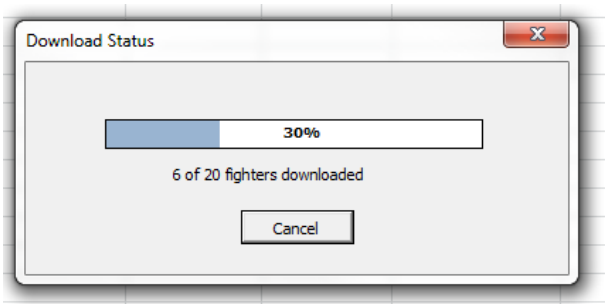


The user should click on the “Download Fighters” button. The user is then presented with a dialog box that prompts them for the fight metric IDs that they would like to download.



The user enters the first and last IDs in the range that they would like to download. The excel sheet will download every fighter including and between those IDs.

The user is then presented with a visually pleasing download bar so they know exactly how long it will be until the web excel document is finished loading the fighters’ statistics. This may take a minute.



If the user decides they have downloaded everything required, they may cancel the download at any time.

Once the fighters have been downloaded, the user will be presented with 20 different data fields for each fighter including height, weight, stance, record, strike accuracy, and take down defense. Please note that not every data field is available for each fighter. Some of the statistics on fightmetric.com are incomplete for less popular fighters.

Below is a screenshot that displays 20 fighters and some of their statistics.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	
1	ID	Name	Nickname	Height	Weight (lbs)	Reach (inches)	Stance	Age	Wins	Losses	Draws	No Contest	Strikes Landed per Minute	Strike Accuracy	Signific
2	1900	Jake Hecht	HITMAN	6' 0"	170	73	Orthodox	29	11	4	0	0	3.26	48%	
3	1901	Willie Parks	SLICK	5' 10"	185	--	--	29	5	2	0	0	--	--	--
4	1902	Siyar Bahadurzada	THE GREAT	5' 11"	170	72	Orthodox	28	21	5	1	0	0.51	29%	
5	1903	Ednaldo Oliveira	LULA	6' 5"	238	81	Orthodox	29	13	1	1	1	0	0%	
6	1904	John Dodson	THE MAGICIAN	5' 3"	125	66	Orthodox	28	14	6	0	0	3.44	34%	
7	1905	TJ Dillashaw		5' 6"	135	68	Orthodox	27	7	1	0	0	4.82	58%	
8	1906	Diego Brandao	CEARA	5' 7"	145	64	Orthodox	25	17	8	0	0	2.19	46%	
9	1907	Dennis Bermudez	THE MENACE	5' 6"	145	66	Orthodox	26	10	3	0	0	5.11	51%	
10	1908	Dustin Pague	THE DISCIPLE	5' 9"	135	74	--	25	11	7	0	0	2.09	49%	
11	1909	John Albert	PRINCE	5' 8"	135	68	--	26	7	4	0	0	4.44	51%	
12	1910	Johnny Bedford	BRUTAL	5' 10"	135	71	Orthodox	30	19	9	1	0	6.46	58%	
13	1911	Louis Gaudinot	GOODNIGHT	5' 3"	125	63	--	28	6	2	0	0	3.25	44%	
14	1912	Roland Delorme		5' 9"	135	71	--	29	8	1	0	1	3.97	50%	
15	1913	Marcus Brimage	THE BAMA BEAST	5' 4"	145	71	Southpaw	28	6	2	0	0	5.1	36%	
16	1914	Stephen Bass	BIGFISH	5' 9"	145	71	--	30	10	1	0	0	4.87	52%	
17	1915	Josh Ferguson	TAZ	5' 5"	125	66	Orthodox	24	8	5	0	0	2.13	44%	
18	1916	Steven Siler	SUPER	5' 11"	145	70	Orthodox	26	21	10	0	0	3.77	43%	
19	1917	Josh Clopton	THE GENTLEMAN	5' 7"	145	68	--	31	6	1	1	0	2.87	69%	
20	1918	Reza Madadi	MAD DOG	5' 11"	155	73	Orthodox	34	13	3	0	0	3.39	57%	
21	1919	Khabib Nurmagom	THE EAGLE	5' 10"	155	70	Orthodox	24	19	0	0	0	2.82	39%	
22	1920	Eddie Mendez		5' 10"	205	--	--		7	0	1	1	--	--	--
23															

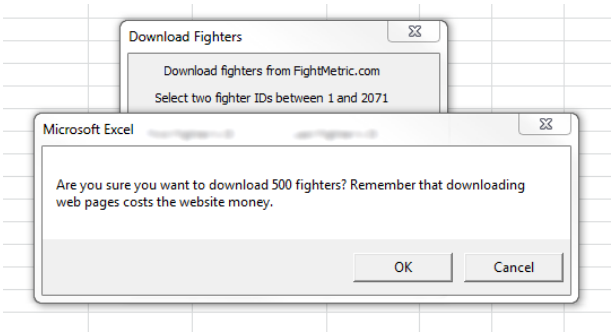
Automated data collection

I understand that automated web requests to a site can generate significant traffic on a web server. While I want the excel sheet to be useful, I wanted to limit the strain on Fight Metric's servers. Fight metric is a database hosting company. It makes much of its data available for end users for free, but it earns all of its revenue from companies that it hosts data for. By downloading the data I recognize that users will not need to go to fightmetric.com, and in a way I am reducing their traffic. However, I feel that this is acceptable because it does not hurt fight metric's revenue model. They have no advertising on their site.

I did take some additional steps to ensure that the excel web requests were acceptable and to limit any unneeded traffic.

First I checked the site's robots.txt file, and the directory that I am crawling allows any user-agent. This includes automated programs.

Second, I added a confirmation dialog if the user attempts to download too many records. Users are free to download the records, but the program does remind the user of the cost.

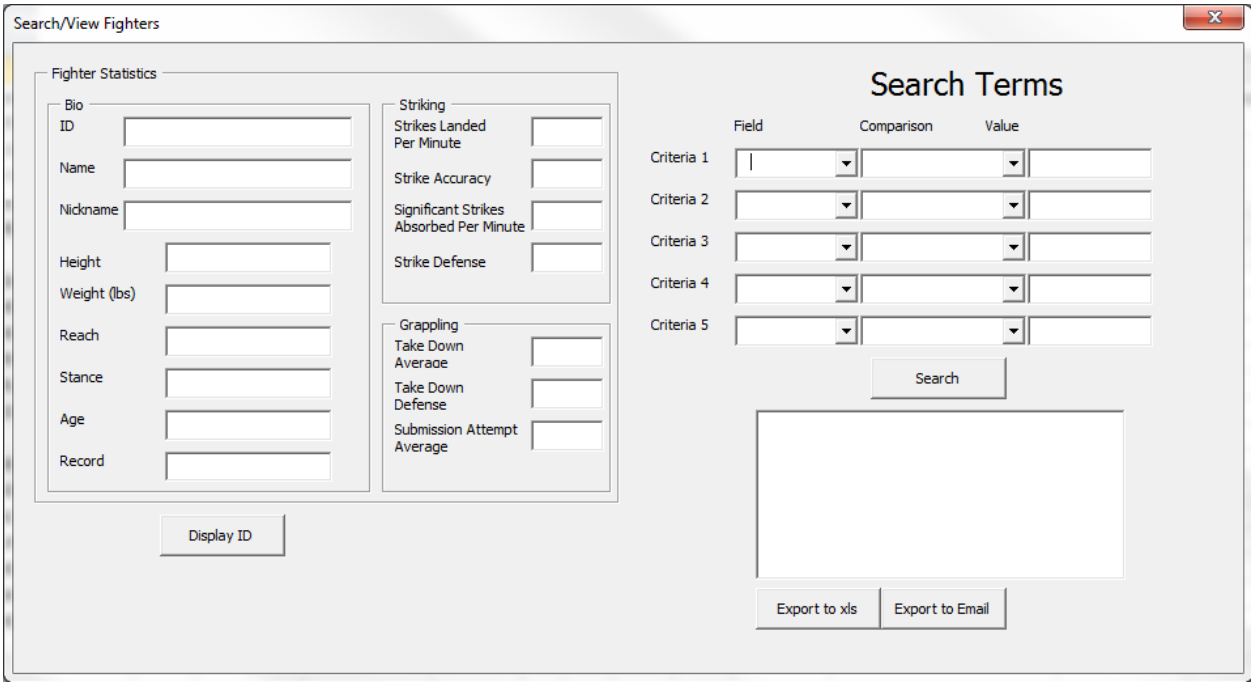


Finally, I implemented a cancel button on the download form. This allows a download process to be aborted if it isn't needed anymore. This is also a convenient feature for the user.

Searching

The second main component of the project is the advanced search form. Users can access the search form by clicking on the "Search for fighters" button on the custom ribbon.

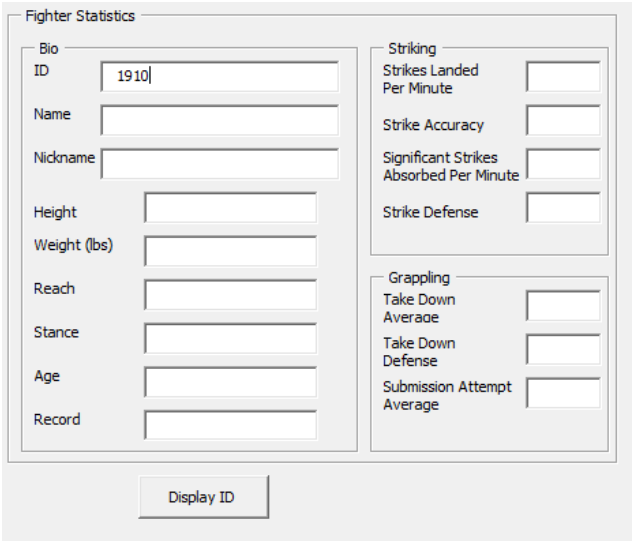
Below is a screenshot of the search user form.



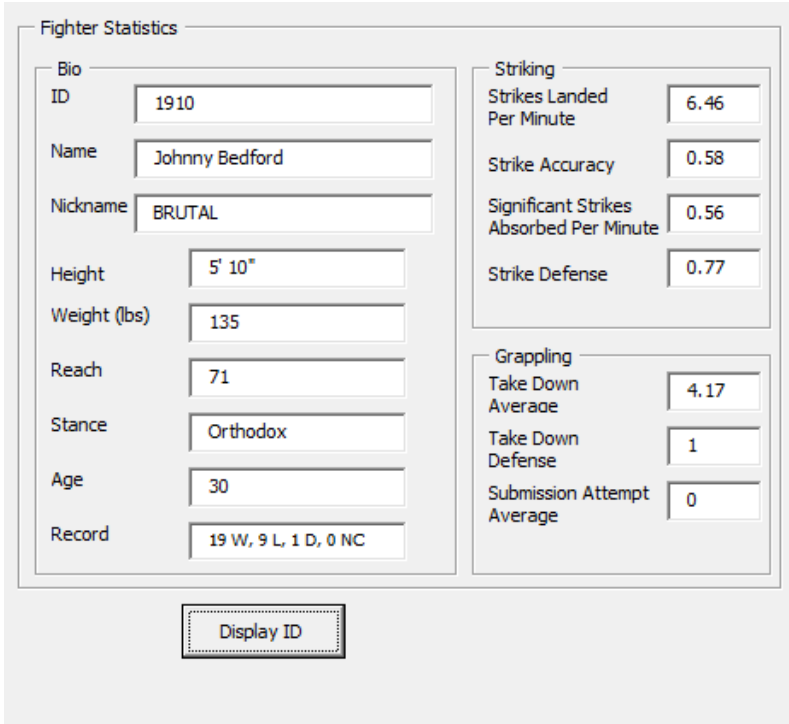
This is the primary interface for a user of this excel document. If someone wanted to they can interact directly with the data in the excel sheet. However, unless a user wishes only to copy a few records or fields, the searching user form provides better functionality.

Because the excel document is intended to download statistics about fighters, it was not designed to allow editing as this would distort the results. The excel document is editable, but the user form does not provide that functionality.

A user can start by viewing a single fighter. This can be done by entering in the desired fighters ID into the appropriate field and clicking the “Display ID” button as shown in this screenshot.



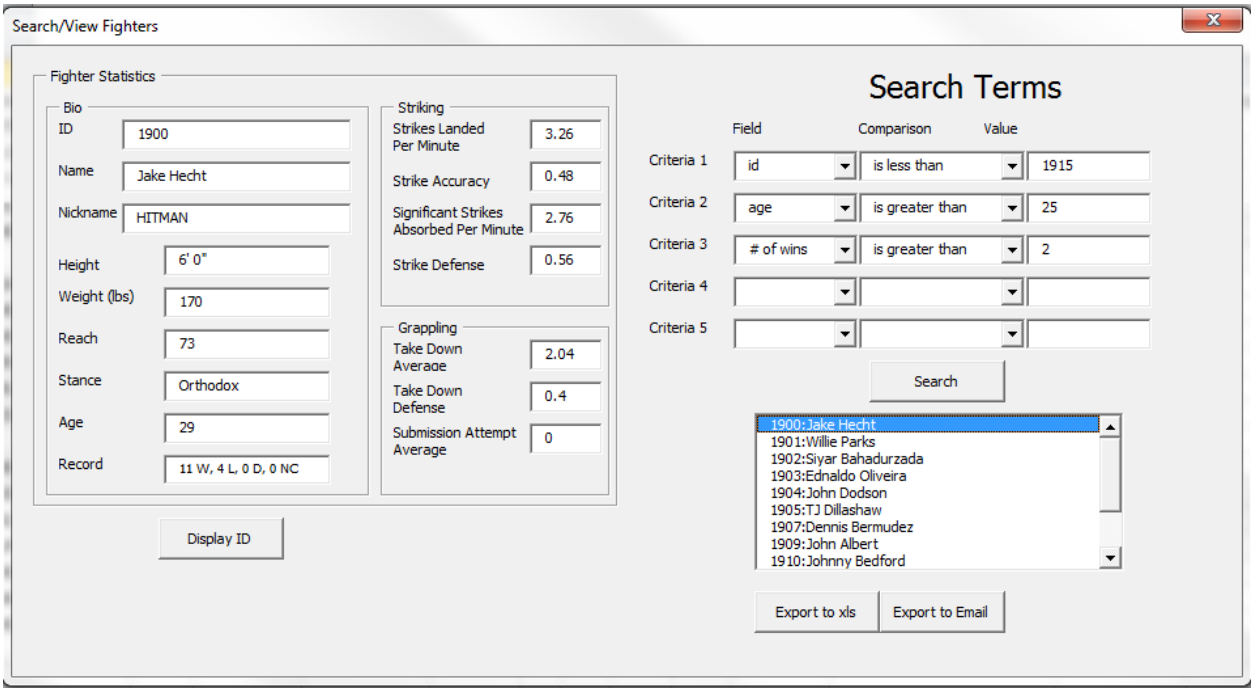
After pressing the button, the user will see all of that fighter’s statistics.



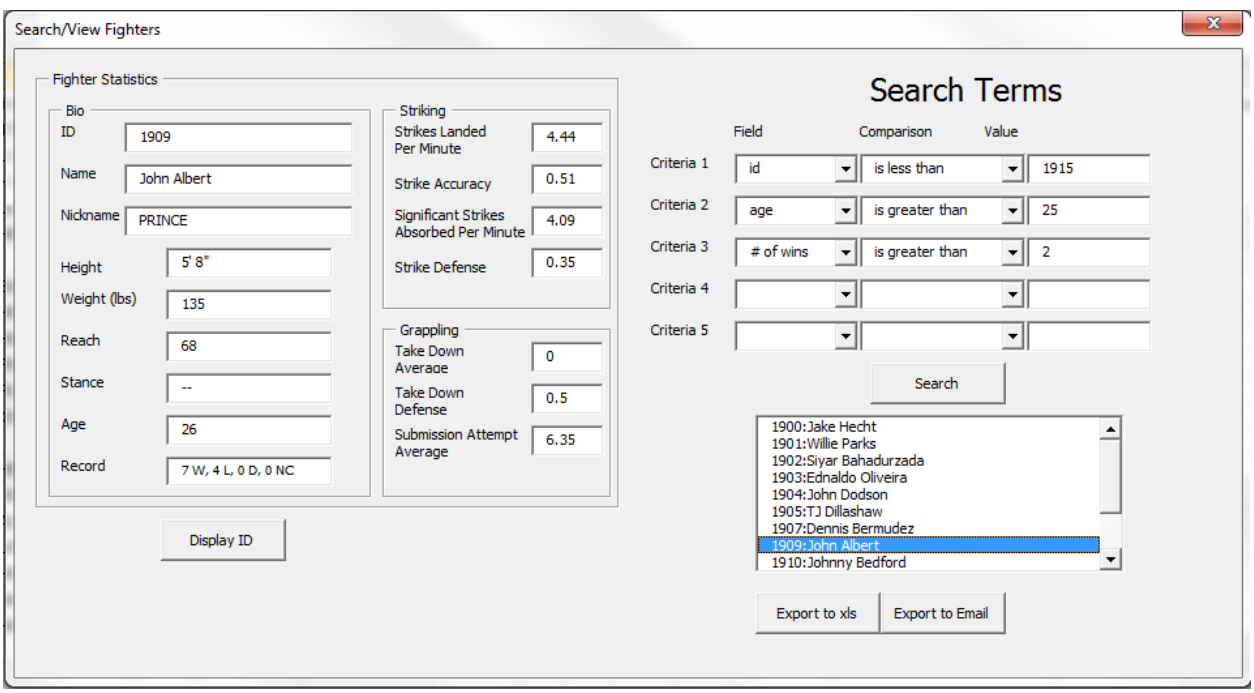
Viewing the entire fighter's details in the form at a glance is useful, but the search function is what makes the form worthwhile.

The form currently supports up to five criteria to search on. A user can use as many or few as desired. Unlike the project 4 search function, this search allows users to input comparator functions. Currently, users can search on 6 fields (id, weight, age, wins, losses, and draws) and can use three logical operators (less than, equal to, and greater than). Once the search button has been pressed, a list box beneath the search fields will be populated with the IDs and names of all the fighters who match the desired filters. Criteria that do not have all of the data entered are ignored.

Below is an example search.



If a user clicks on one of the entries in the list box, it populates that fighter's data into the statistics fields on the left. Notice that the fighter John Albert is now selected.



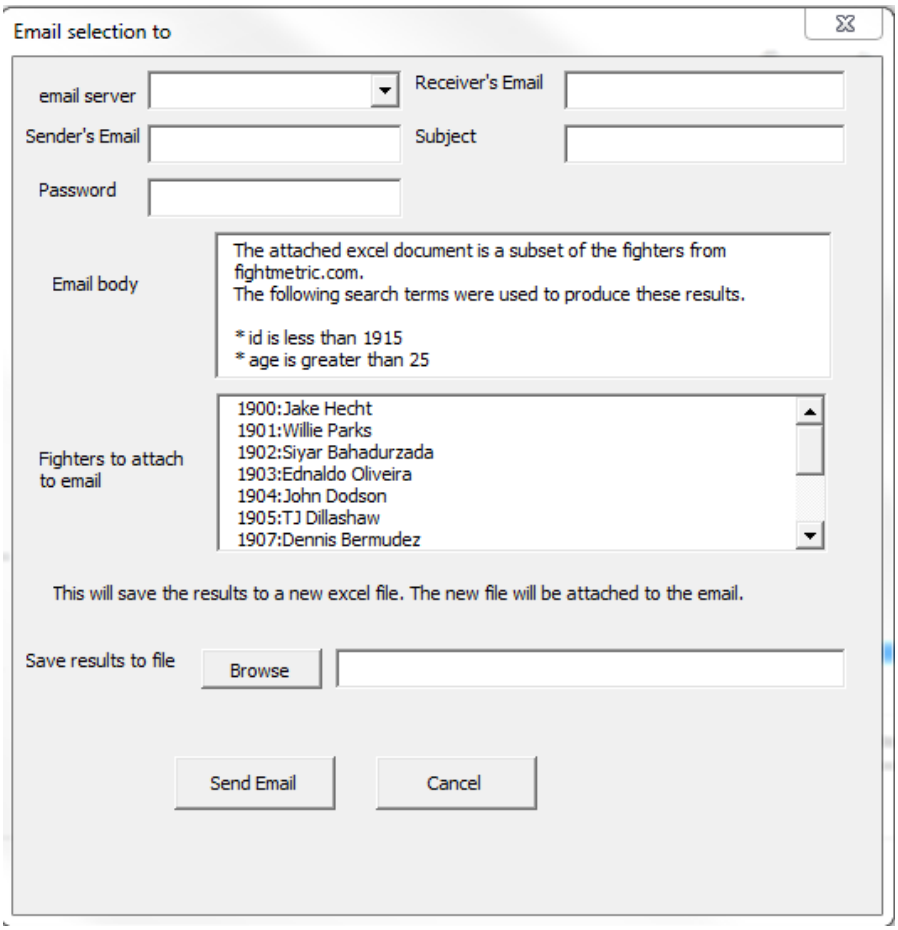
Once a user has refined their search down to the correct subset, it is time to export it.

Export Data

The worksheet provides two ways to export search results. They can be sent either to another worksheet, or sent as an attachment in an email

To save the search results in another workbook, the user just needs to click the Export to xls button. It will open a file dialog box, create a new workbook, and then copy the search results and the header into the new workbook.

The second way to export the data is in an email. To do so, a user needs to click on the "export to email button". This will open the email user form shown below.



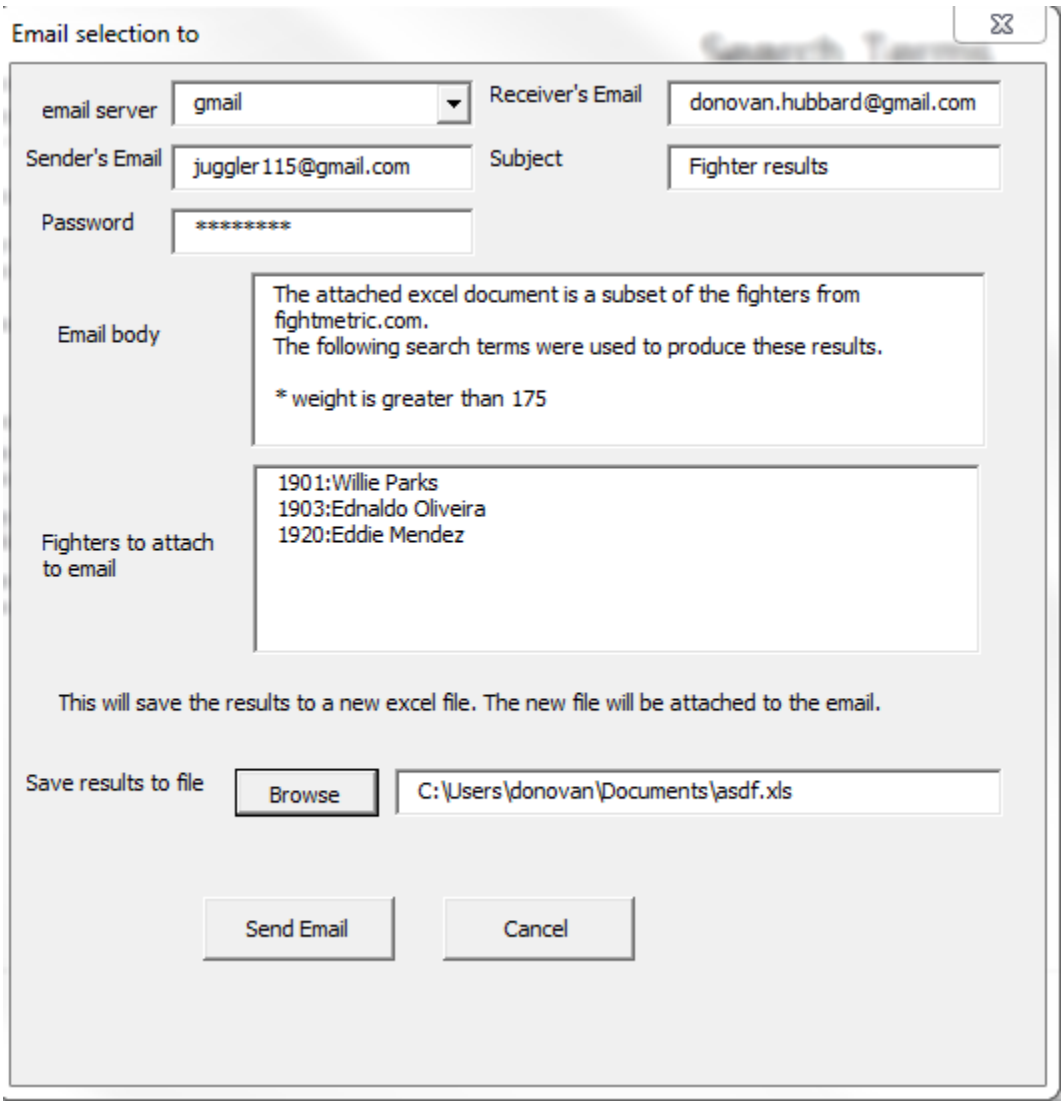
The email form has several important fields. First, it has a combo box that prompts the user to enter their email server. Currently this excel document only supports sending emails through Gmail or Yahoo. Next the user needs to input their email address and password. Then the user selects the receiver's email and the subject line the email should use.

The next two text boxes contain the information that will be sent in the email. The first text box is labeled email body. This is the message that will be sent in the main email body to the recipient. This

can be modified by the user. By default it includes a basic description of the contents, including a list of all of the filters used to search the spreadsheet with.

Finally, the user form needs to save the results to a workbook that will be attached to the email. Clicking the browse button will open a file dialog box that selects the file name to store the workbook in.

Please note that the emailing functionality requires the “Microsoft CDO for Windows 2000 Library” add-in.



The user completes the transaction by clicking the send email button. If everything went well, the user will receive a message box that informs them of the success.

Learning and conceptual difficulties encountered

This project includes the following in-class concepts

- Scraping web pages with the agent class
- Heavy use of user forms
- Exporting data to different worksheets
- Emailing worksheets directly from excel.

Web automation

One of the most valuable things that this assignment let me experience was learning how to automate complicated web scraping. There is so much data online that could be useful, but it can be difficult to aggregate it all together.

One of the challenges with this spreadsheet was to effectively gather the data. I had to find how the data was organized. The web directory that I am crawling can be reached at the following URL.

http://hosteddb.fightmetric.com/fighters/details/<fighter ID>

Once I load the site, I can parse through the data by reading all of the div tags. One thing that I noted is how some of the data was much easier to identify than others because of how specific or generic the div tags were. I have determined that if I were to be a web developer I would want to have fairly unique tags for most of the data in order to allow people to search the data more effectively.

One of the big challenges that I encountered with web scraping is that sometimes the data was not rendered consistently across all of the pages. I had to include error handling in order to deal with the exceptions that occurred.

User forms

This project had a total of four user forms, and most were fairly complicated. Variable scope made a big difference as I was passing data from one form to another. I had to make sure that the correct variables and functions were public.

I was also impressed by the progress bar form that Professor Allen used on one of the projects. I dug into the code to see how he implemented it and found that VBA has no download bar control. By

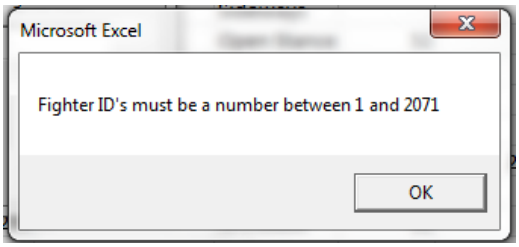
mimicking Professor Allen's code, I was able to combine several different form controls, a label and two text boxes of different colors, to create a convincing download bar.

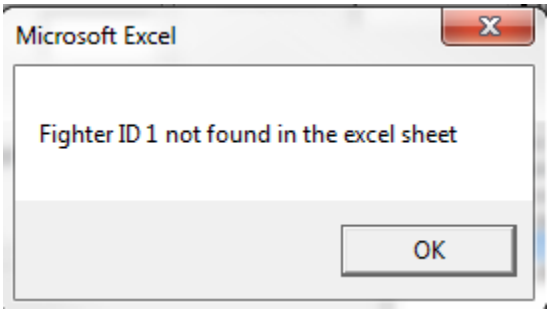
Constants

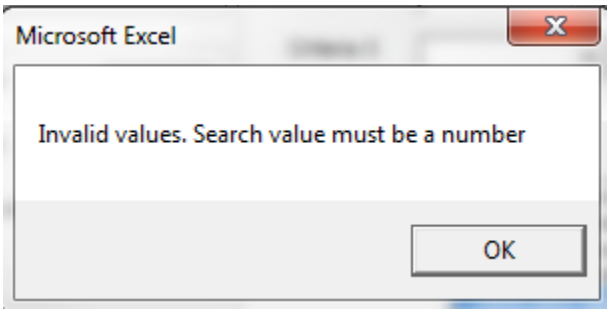
I used quite a few constants in this project. All of the spreadsheet columns had constants assigned to them. When I wanted to print something in a particular column, I would just reference the constant. This came in handy because there were several times when I changed the order that the columns were being printed in. If I would have hard coded the values than it would have been very difficult, but since I used global variables I only had to change a few lines of code.

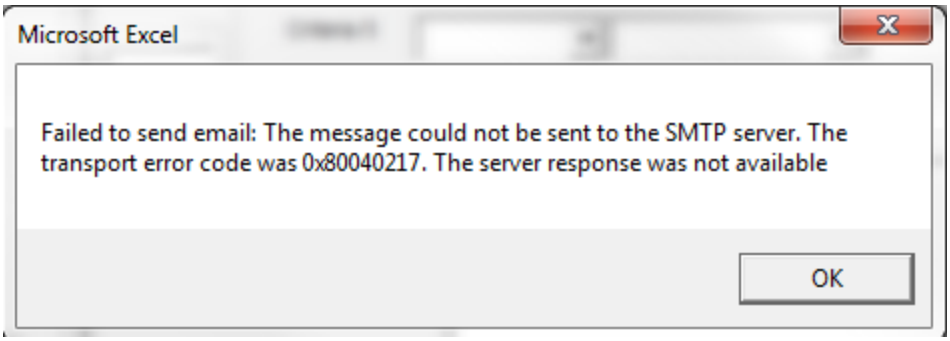
Data validation

Data validation takes a long time, but it is worth it. I had to write quite a bit of code in order to handle erroneous data that a user can enter. However, without the error handling the code will break. Users will enter incorrect data and not realize what they did wrong. Below are some of the error messages that are displayed to the screen if incorrect input is submitted.









As far as user forms are concerned, you have to validate everything in order to help the user and to keep your code from braking.

Timing

The last lesson that this project stressed to me was that everything always takes longer than you thought it should have. There were quite a few features that appeared to be simple and straightforward but required two hours of work to get down properly. This is something really useful to keep in mind the next time my boss asks me to program something in excel.

Unimplemented features

There are several features that I would have liked to include, but I didn't have enough time to complete them.

The biggest one is that I would like to allow users to dynamically add more search criteria. Currently there are five on the form, but I could add more without too much difficulty. I was originally planning to do this and the code is written in such a way that adding another would not be dreadful. All of the fields follow are certain naming convention and I can iterate over them with a single variable use as a counter. Since this variable is just a global integer, I could add or subtract that value as needed and iterate over a new number of controls.

For example, in a for loop I reference the controls with a line of code that looks similar to this.

currentValue = Me.Controls("comboComparator" & i).value

I hadn't determined the best way to add new fields yet. I was either going to create them the fly, or to have them created already, but just hidden.

Another relatively simple feature to add would have been more email domains. I only need to figure out what url I need to use in order to access the smtp server.

I also wanted to make more fields searchable. Adding numeric fields would have been easy because the code is already setup to handle that. The difficult part would have been to differentiate between text or numeric fields or even worse, height.

I had been brainstorming ways to search on height, and the best way that I came up with was to convert the height into inches. This would have taken some time to come up with a correct parsing algorithm, but I think it would have added significant functionality.

Conclusion

This was a good project that took significantly longer to complete than I had originally intended. It helped cement in a lot of concepts that I had covered in class but not committed to memory. It also helped to remind me of some of the higher principles involved in programing. There were a few times when I decided to do something the quick and dirty way and it came back to bite me later. Code reuse and using variables instead of hard coded variables made adding new features a lot simpler.