How to search Completed Items Using the eBay API

**GETTING STARTED**

Warning: What I am about to teach you is illegal in 49 states…just kidding. But seriously, you must be careful how you use the completed auctions data in eBay’s API as you could be violating their terms & conditions.

STEP 1 – Sign Up

Sign up for eBay’s Developer program. This is easy to do by clicking on the Sign In/Join link at the top of the page at: <https://go.developer.ebay.com/>

[SCREENSHOT]

STEP 2 – Get your Key

Congratulations on completing the easy part!

For this tutorial, we will be using the HTTP GET call method which then returns the eBay data in XML format. In order to do this, you will need your AppID Production Key. From the main eBay Developers Program homepage, underneath Get Started click the link that says Get your keys.

[SCREENSHOT]

It may seem counter-intuitive, but it in order to test your API calls against actual eBay products, you will need your Production Keys NOT the Sandbox Keys. Click the Generate Production Keys button. Once the keys are generated, under Production Keys, copy your AppID into a text editor (or memorize it if you are EXTREMELY ambitious and have brain space to waste) as you will need this to make your first API call.

STEP 3 – Your first API Call

It’s time to make your first API call. In the URL below, paste your AppID where the asterisks are (there should be no asterisks once you have pasted in your AppID), then copy and paste the entire URL into the browser.

[http://svcs.ebay.com/services/search/FindingService/v1?OPERATION-NAME=findCompletedItems&SERVICE-NAME=FindingService&SERVICE-VERSION=1.13.0&GLOBAL-ID=EBAY-US&SECURITY-APPNAME=\*\*\*\*\*\*\*\*\*\*\*\*\*&RESPONSE-DATA-FORMAT=XML&REST-PAYLOAD&categoryId=11970](http://svcs.ebay.com/services/search/FindingService/v1?OPERATION-NAME=findCompletedItems&SERVICE-NAME=FindingService&SERVICE-VERSION=1.13.0&GLOBAL-ID=EBAY-US&SECURITY-APPNAME=*************&RESPONSE-DATA-FORMAT=XML&REST-PAYLOAD&categoryId=11970%20)

When working correctly, you should see something like this:

[SCREENSHOT]

**API CALL STRUCTURE**

Anatomy of an API Call

Let’s take a look at the call we just made in order to better understand what is going on:

<http://svcs.ebay.com/services/search/FindingService/v1> - This is called the Service Endpoint and dictates which API you are calling. In this case, we are calling the eBay Production API because we want to get results from the live eBay site. If you wanted to use the eBay Sandbox instead you would need to change this part of the URL.

OPERATION-NAME=findCompletedItems – The operation-name parameter is used to define what kind of call you are using. In this case, we are concerned with finding completed items, but there are many other options you could choose from.

SERVICE-NAME=FindingService – The service-name parameter specifies which API service you are using. We are using the Finding API but other options include, but are not limited to, Shopping API, Feedback API, Trading API, etc.

SERVICE-VERSION=1.13.0 – This is simply the version of the API you are working with. Typically you want to use the most up-to-date version as possible in order to take advantage of all of the functionality. As of this writing, the most up to date version is 1.13.0

GLOBAL-ID=EBAY-US – This value dictates which site your information will come from. In our examples, we will be using the regular US version of ebay.com but you could change this value to find information for any of eBay’s sites.

SECURITY-APPNAME= - This is the parameter where you paste your AppID so eBay can trace all of your API calls. There are limits to how many calls you can make, so be careful.

RESPONSE-DATA-FORMAT=XML – This parameter specifies the format of the response. In our examples, we will be using XML but NV, JSON and SOAP are also available.

REST-PAYLOAD – This value simply separates one part of the URL from the other. All of your custom parameters will come after this value.

categoryId=11970 – In order to make a call to findCompletedItems, you must include keywords or a categoryId value. In this case, we included the categoryId value of 11970 which is the Seated Half Dollars category. A complete list of categorIds can be found here: http://www.isoldwhat.com/getcats/

**More on Parameters/Filters**

When making an API request, we want to make sure we only get the exact information we want. Including information that is unnecessary not only slows down the request, but makes the processing of data on the backend that much more difficult. In order to limit what is returned, we use filters. The two most effective filters to use are the item and aspect filters.

ITEM FILTERS

Item filters are data points that are common to all eBay items. These include item condition, MaxPrice, FreeShippingOnly, etc. To narrow our search results, let’s add the name item filter. This will allow us to limit our completed items results to just the ones that actually sold. We do this by adding the following to the end of our API call:

&itemFilter(0).name=SoldItemsOnly&itemFilter(0).value=true

You will notice that I included two item filters. The first defines the actual filter name while the second determines the value. In this case, we only want to see items where SoldItemsOnly = true. You can include as many item filters in an API call as you want. Simply increase the number next to itemFilter to designate a different filter. For example:

&itemFilter(0).name=SoldItemsOnly&itemFilter(0).value=true&itemFilter(1).name=Condition&itemFilter(1).value=New

Our master API call should now look like this:

<http://svcs.ebay.com/services/search/FindingService/v1?OPERATION-NAME=findCompletedItems&SERVICE-NAME=FindingService&SERVICE-VERSION=1.13.0&GLOBAL-ID=EBAY-US&SECURITY-APPNAME=SLcoinafa-2536-4ac4-af19-68edaf5f9f6&RESPONSE-DATA-FORMAT=XML&REST-PAYLOAD&categoryId=11970&itemFilter(0).name=SoldItemsOnly&itemFilter(0).value=true>

ASPECT FILTERS

Aspect filters are data points that are specific to a category on eBay. These are also known as item specifics and are usually found in the left column of the search results. When searching for Computers, you may see filters such as RAM, CPU Clock Speed, Ports, etc. These are all things that would be specific to computers but would not apply to Men’s Clothing, for example.

Aspect filters are a great way to further refine our search results to make sure we are only getting the results we want. In our example, we now have a result set that includes completed items in the Seated Half Dollar category that have been sold. Let’s say I want to only see sold coins that have been graded by PCGS. I would add the following to our API call:

&aspectFilter(0).aspectName=Certification&aspectFilter(0).aspectValueName=PCGS

Just like with item filters, we can include as many aspect filters as we need. I will include sold coins that have been graded by NGC as well:

&aspectFilter(0).aspectName=Certification&aspectFilter(0).aspectValueName=PCGS&aspectFilter(1).aspectName=Certification&aspectFilter(1).aspectValueName=NGC

Our complete API call now looks like this:

<http://svcs.ebay.com/services/search/FindingService/v1?OPERATION-NAME=findCompletedItems&SERVICE-NAME=FindingService&SERVICE-VERSION=1.13.0&GLOBAL-ID=EBAY-US&SECURITY-APPNAME=SLcoinafa-2536-4ac4-af19-68edaf5f9f6&RESPONSE-DATA-FORMAT=XML&REST-PAYLOAD&categoryId=11970&itemFilter(0).name=SoldItemsOnly&itemFilter(0).value=true&aspectFilter(0).aspectName=Certification&aspectFilter(0).aspectValueName=PCGS&aspectFilter(1).aspectName=Certification&aspectFilter(1).aspectValueName=NGC>

Handling the XML Output

Now that we have our API call setup how we want it, next we have to deal with the XML output that eBay’s system responds with. For this example, we will be using a simple PHP program to show how to break down the mass of data returned in the XML.

Here is the simple PHP program:

A quick breakdown of what is going on here.

Line 2 – The simplexml\_load\_file function turns eBay’s XML response into a manageable object we can reference. All of the data will be stored in the $xml variable.

Line 4 – We need to check if our call is successful by checking if ack equals Success. Any other response and things are not working as they should be and an error message will be printed.

Lines 6-12 – Here we create a $results variable where we will be storing all of the individual data points. We then use a foreach loop to step through every individual item contained in the Search Results XML response. At this point, you can access any of the item parameters, but for this example we will just take the title and save it in our results variable. Once the loop is complete, the program will print out a simple table with the titles for every sold half dollar that has been graded by PCGS or NGC.

Saving the XML data to your database

At this point, all of the heavy lifting is done! You can rest easy, knowing that in a few short minutes you will have a database full of completed eBay items. Just a reminder, this is for educational purposes only. Saving completed item data is a violation of eBay Terms & Conditions. You have been warned.

Full tutorials on creating, editing and managing MySQL databases are available all over the internet so I will spare you those details. Instead, I just wanted to flesh out a quick tweak on our previous PHP program that will allow you to store the item data instead of print it out on the screen.

[SCREENSHOT]

You can see in this example that we open up a connection to our database and then within the foreach loop, we define the item variables we wish to save and then save them into our database. With each loop a new item is being added to the database until there are no items left.

And there you have it! By following these simple instructions, you can use the eBay API to create a quasi-legal database full of market information that you can capitalize on to become a multi-millionaire. When you do, make sure and donate some of that money to Oregon State since that is where you learned this nifty trick!