

Consuming Amazon.com E-Commerce

An overview



By Jeff Houser

One of the most popular articles I've ever written was on Amazon.com Web services. The article was written for the Macromedia Developer Center and is located at [http://www.macromedia.com/](http://www.macromedia.com/devnet/coldfusion/articles/wsamazon.html)

<http://www.macromedia.com/devnet/coldfusion/articles/wsamazon.html>.

This article is a bit dated. A few days before it was published, Amazon.com released version 1 of their SOAP Web services. In this issue on exchanging data, I thought it'd be great to write an article on version 4 of Amazon.com's SOAP interface, released in September. They are now called "Amazon.com E-Commerce Services."

After two days of much frustration, I was unable to successfully invoke the SOAP Web services from ColdFusion or BlueDragon; I kept getting semantic errors when compiling the WSDL. However, you can still use Amazon.com Web services without SOAP, using REST instead, which is a SOAP alternative. In this article, I'll introduce you to Web services and show you everything you need to know to implement Amazon.com's E-commerce services on your site.

Web Service Definitions

Before delving into Amazon.com specifics, I want to make sure we're all on the same page when it comes to terminology. If you're already familiar with Web services, you can probably skip this section. If you are new, or want a reminder primer, read on.

Most of the time when you hear someone talk about Web services, they're talking about SOAP. SOAP stands for Simple Object Access Protocol, an XML dialect that is used to define how Web services can communicate with the programming language calling it. Not all Web services are SOAP, though. A Web service is defined as any program run over a network by another program. If you've ever used cfhttp to process a credit card transaction through a gateway provider,

you've used a non-SOAP-based Web service. These types of requests are sometimes called REST requests. REST stands for Representational State Transfer. Instead of passing an XML-based SOAP request, REST is done by using a variety of URL parameters to specify which methods to invoke and which values to pass in. More information on REST can be found at http://www.ics.uci.edu/%7Efielding/pubs/dissertation/rest_arch_style.htm.

The next definition is the WSDL, the Web Services Definition Language. It defines the functionality of a SOAP Web service. WSDL is another XML variant. The WSDL will tell you the operations you can perform against the Web service and what you would expect to get in return for that service. In many cases, you'll be provided with English documentation (such as a Word doc, PDF, or HTML) for the Web services and won't have to read the WSDL manually. If you think of a Web service as a CFC, the WSDL will define the names of methods, the input parameters to those methods, and the type of data that is returned from those methods. If you create a Web service in ColdFusion, the WSDL is generated automatically for you, just by appending "?wsdl" to the URL. In ColdFusion, you'll need to know the location of the WSDL is to create a Web service object.

There are two other definitions that you can use when speaking about Web services. The first is to publish a Web service. When you publish a Web service, you're making it available for people to use. If the Web service is public, such as Google or Amazon's Web services, anyone can use it. If the Web service is private, you may only be providing access to clients, customers, or other departments in your company. The act of publishing is merely telling someone the URL and providing the documentation on how to use the Web service.

On the other side of the coin, Web services are consumed. To consume a Web service means that you are using a Web service. It's as simple as that. Amazon.com published their Web services by making them public. As the one using the Web service, we are consuming them. Publish always seemed like a logical name for making Web services available, but it took me a while to get used to "consuming" them.

Getting Started with Amazon.com Web Services

Why would you want to use Amazon.com Web services?

Perhaps you're an associate and want to bump up your potential for income by allowing people to search the full Amazon catalog. Perhaps you're a merchant who is selling a product through the Amazon.com store. The Amazon.com e-commerce services offer many features, including:

- **Product Searches:** If you take away its product catalog, there isn't much to the Amazon.com Web site. You have the full array of search options available through the Web service interface that you do via the Amazon.com Web site. After a search, you can always get full details on an individual product.
- **Product Images:** Image data is returned as part of the search.
- **Customer Reviews:** You have full access to the customer reviews for any given product.
- **Wish List Searches:** You can search wish lists by name, e-mail address, city, or state using the Web service interface.
- **Remote Shopping Cart:** You can now have your users add items to their Amazon.com shopping cart without leaving your own site.

To get started setting up Amazon.com Web services on your site, first register for a subscription ID. You can do so at the Amazon.com site: <http://www.amazon.com/gp/browse.html/104-7075718-2365513?node=3435361&/104-7075718-2365513>. If you're not an Amazon.com associate (but want to be), you can join the associate program here: http://www.amazon.com/gp/browse.html/ref=gw_bt_as/104-7075718-2365513?node=3435371.

Deconstructing the Amazon.com URL

You're ready to perform your search. The first step is to construct your URL. The URL starts with the location <http://webservices.amazon.com/onca/xml>. It is the query string parameters that provide us with the search results:

- **Service:** The service parameter accepts the value of AWSECommerceService to access Amazon.com E-commerce services. It is required.
- **SubscriptionId:** The subscriptionID was provided to you when you signed

up for the developer program. This is also a required parameter.

- **Operation:** The operation parameter describes what operation you are trying to perform. There are 18 different actions that you can perform. In this article I'll show you ItemSearch and ItemLookup. You can also look up seller info, perform similarity searches, add information to the user's Amazon.com shopping cart, or look up public customer information (such as reviews or wishlists). This is required.
- **AssociateTag:** If you are an Amazon.com associate, put your associate tag here to ensure that you keep getting the kickbacks when people buy something based on your referral. This is optional.
- **ResponseGroup:** The responsegroup parameter is used to define the type of information that is sent back to you
- **Operation Parameters:** The operation parameters are a set of parameters required for each operation. You might think of this as similar to the way that cfile works. You use a different set of parameters to upload a file than you do to copy a file, move a file, and so on. Other tags, such as cfoject and cfloop, operate in a similar manner.

Note that all of these parameters are case sensitive.

Searching the Store

Now you want to search the Amazon.com store. The operation that you want to perform on the store is ItemSearch. These are some ItemSearch-specific parameters:

- **SearchIndex:** The search index specifies the name of the Amazon.com store index that you are searching. Available indexes are locale specific. Some common ones are Books, Music, DVD, and Software. A complete list is available in the Amazon.com DVD. If you set this value to blended, all indexes will be searched. This is required.
- **Keywords:** The keyword search will compare your search term against various product fields, including (but not limited to) title, author, artist, and description.
- **BrowseNode:** The browsenode parameter is used to narrow your search to a specific category of products in the Amazon.com catalog. You can think of browse nodes as subcategories within each SearchIndex. For example, to find ColdFusion books, you would probably search in Professional & Technical. This is an optional attribute.
- **Availability:** This is an optional parameter that specifies that you want to only search for items that are available. If specified, you want to give it the value of Available.
- **Title:** The title parameter is optional and used to search for a specific title. In addition to the title, you can search using Author, Publisher, Artist, Actor, Director, Manufacturer, MusicLabel, Brand, and a whole slew of others. Check the documentation for the complete list.
- **MinimumPrice:** The minimum price is used to specify a price range. Items that are priced lower than this value will not be returned. Its parallel MaximumPrice can be used to specify an upper limit, where items with prices higher than this value will not be returned. If either is left out, there is no lower or upper limit, respectively.
- **Condition:** The condition parameter is used to filter the list using the specified value. Valid values are All, New, Used, Refurbished, or Collectible.
- **Sort:** Specifies the sort order of the result, similar to the "Order By" clause in SQL. Its values are dependent upon the locale; in the U.S. you can sort using these values: daterank, relevancerank, pricerank, -titerank, inverse-pricerank, and salesrank. The minus uses a descending sort, where applicable.
- **ItemPage:** ItemPage is used to paginate the results. The search will return only 10 results. You can use ItemPage to specify which page of search results you want. The default is to return the top 10 results. But, if you specify two, you'll get results 11–20; if you specify 100, you'll get results 100–110, and so on.
- **ResponseGroup:** The ResponseGroup parameter is intended to provide greater flexibility on the type of data

that is returned. In previous versions of Web services, you could only retrieve data in two formats: light (some data) or heavy (all data). Response Groups are designed to give you much flexibility in what is returned. If you wanted to perform a search that only returned image data and the title, then response groups are for you.

Okay, you're digesting a lot of information. Let's see some code:

```
<cfhttp url="http://webservices.amazon.com/onca/xml?Service=AWSECommerceService&SubscriptionId=#variables.SubscriptionID&Operation=ItemSearch&SearchIndex=Books&Keywords=ColdFusion&Author=Houser" method="get">
</cfhttp>
```

```
<<cfdump var="#XMLParse(cfhttp.FileContent)#">
```

To figure out what's going on, you'll need to dissect the URL. You know the service value is constant. The subscriptionID was provided by Amazon.com (in this case, the value is stored in the variables scope). The operation is ItemSearch. The SearchIndex is Books, which means we are searching the Amazon book catalog. The keyword that the code searches for is ColdFusion. The author is Houser (me). The final line of the code dumps the results of our query string. Since cfhttp returns a string, before dumping, the XMLParse function is used. It makes the dump look nicer. After all the preparation, the retrieval is really that simple. You can easily make a form to allow the user to enter parameters as necessary.

Retrieving One Item with ItemLookup

You're going to process the search results and your user will want to get more details on an item. You can get

them using an ItemLookup. These are some common parameters:

- **ItemId:** It specifies the product ID of the value you want to look up. If you want to retrieve multiple IDs, you can do so by adding a comma-separated list of the products you want to return. Most likely you'll use an ASIN number (Amazon's internal ID) to get information on the specific product, but other options are available such as SKU and UPC.
- **IdType:** This parameter is used to specify which type of value is contained in the ItemId parameter. The default value is ASIN, but you can also specify SKU, UPC, or EAN.
- **SearchIndex:** It specifies which amazon.com store you are searching for your item in. If you use the ASIN, this is ignored. If you are performing an alternate search, this is required. It operates the same way as SearchIndex does for the ItemSearch.
- **ResponseGroups:** Response groups here act the same way that response groups do in an ItemSearch. In the following example, I actually used a simple response group to retrieve more data than just title and author.

```
<cfhttp url="http://webservices.amazon.com/onca/xml?Service=AWSECommerceService&SubscriptionId=#variables.SubscriptionID&Operation=ItemLookup&ItemId=0072132388&ResponseGroup=Request,Large" method="get">
</cfhttp>
```

```
<<cfdump var="#XMLParse(cfhttp.FileContent)#">
```

You should already be familiar with the Service and SubscriptionId. The operation is ItemLookup to retrieve an item. The ItemId contains the ASIN of the value to retrieve. The ResponseGroup is set to large, which returns a multitude of information


about the product including image locations, customer reviews, lists that include the book, price, and just about anything you'd want to know about the book.

Where to Go from Here

I've showed you how to get and receive data from the Amazon.com E-commerce interface using ColdFusion. Unfortunately, due to space constraints, there is barely time to scratch the surface of the full scope of functionality. You can examine Amazon.com's documentation, located at http://www.amazon.com/gp/browse.html/ref=sc_fe_1_0_3435361_3/104-7075718-2365513?%5Fencoding=UTF8&node=3487571&no=3435361&me=A36L942TSJ2AJA.

To give you an idea of the full breadth of the service suite, the PDF document is over 450 pages long. You may also look at the Amazon Web service blog at <http://aws.typepad.com/> or talk with other users at the discussion forum (<http://forums.prospero.com/am-assocdevxml>).

This article didn't discuss what to do once you get the XML data returned. You may want to examine ColdFusion's XML handling, so you can actually do something with the data once you have it. You might start with the July 2005 issue of *CFDJ* (Vol. 7, issue 7); most of the articles in that issue focused on XML, including an XML prep in this column. Neat things can be done with XSL and the Web service suite.

Finally, on a personal note, I've officially entered the world of blogging (thanks Ray). Check out my personal site at www.jeffryhouser.com for my ramblings on ColdFusion, business, and a spattering of random thoughts. Until next time! 

About the Author

Jeff Houser has been working with computers for over 20 years and in Web development for over 8 years. He owns a consulting company and has authored three separate books on ColdFusion, most recently ColdFusion MX: The Complete Reference (McGraw-Hill Osborne Media).

jeff@instantcoldfusion.com

“A Web service is defined as any program run over a network by another program”