Amazon E-Commerce Service

ECS 3.0 to 4.0 Migration Guide

ECS 3.0 to ECS 4.0 Migration Guide

Contents

Getting Started

About This Guide

The goal of this guide is to ease your migration from ECS 3.0 to ECS 4.0. This guide highlights the major differences between the versions and shows you how to recreate the same functionality using ECS 4.0 that you used with ECS 3.0. While this guide will provide a basic introduction to ECS 4.0 concepts, you should refer to the full *Amazon E-Commerce Developer Guide* for more details.

If, after reading this guide and consulting the ECS 4.0 documentation, you have more questions, please post them to the ECS Developers Forum at http://developer.amazonwebservices.com/connect/forum.jspa?forumID=9&start=0.

Why Migrate to ECS 4.0?

Moving to ECS 4.0 has a number of benefits over ECS 3.0:

- **Improved data breadth and quality**
  ECS 4.0 provides access to additional data through an expanded set of response groups, operations and parameters. Examples include TopSellers and NewReleases for browse nodes, access to Wedding and Baby registries, and more powerful searching capabilities. In addition, data accuracy and quality has been improved in ECS 4.0.

- **Extended capabilities**
  The features and capabilities of ECS 4.0 have been greatly improved. New operations such as BrowseNodeLookup, CustomerContentLookup and ListSearch allow you to get information about the Amazon product hierarchy, find customer generated content, and search for wish lists. Such features simply don't exist in ECS3.

- **Ongoing enhancements**
  ECS 4.0 is fully supported and under active development by the ECS team. While ECS 3.0 will also continue to be supported until deprecation, it will only be maintained at its current level of service. All new features and enhancements are only being added to ECS 4.0.

This guide will help you gain access to the latest capabilities and most powerful API that ECS has to offer.

Making Requests

Just like ECS 3.0, ECS 4.0 fully supports REST and SOAP interfaces. However, unlike ECS 3.0, you'll find far more consistency between the two – there are no differences between parameter names and expected values in ECS 4.0.

WSDL and Schema Locations

WSDL

Locale	URL
US	http://ecs.amazonaws.com/AWSECommerceService/AWSECommerceService.wsdl
UK	http://ecs.amazonaws.co.uk/AWSECommerceService/UK/AWSECommerceService.wsdl
DE	http://ecs.amazonaws.de/AWSECommerceService/DE/AWSECommerceService.wsdl
FR	http://ecs.amazonaws.fr/AWSECommerceService/FR/AWSECommerceService.wsdl
CA	http://ecs.amazonaws.ca/AWSECommerceService/CA/AWSECommerceService.wsdl
JP	http://ecs.amazonaws.jp/AWSECommerceService/JP/AWSECommerceService.wsdl

XML Schema

Locale	URL
US	http://ecs.amazonaws.com/AWSECommerceService/AWSECommerceService.xsd
UK	http://ecs.amazonaws.co.uk/AWSECommerceService/UK/AWSECommerceService.xsd
DE	http://ecs.amazonaws.de/AWSECommerceService/DE/AWSECommerceService.xsd
FR	http://ecs.amazonaws.fr/AWSECommerceService/FR/AWSECommerceService.xsd
CA	http://ecs.amazonaws.ca/AWSECommerceService/CA/AWSECommerceService.xsd
JP	http://ecs.amazonaws.jp/AWSECommerceService/JP/AWSECommerceService.xsd

Endpoints

When migrating to ECS 4.0, you should update your application to use the appropriate service endpoint.

Interface	Locale	ECS 4.0
REST	CA	http://ecs.amazonaws.ca/onca/xml?Service=AWSECommerceService
	DE	http://ecs.amazonaws.de/onca/xml?Service=AWSECommerceService
	FR	http://ecs.amazonaws.fr/onca/xml?Service=AWSECommerceService
	JP	http://ecs.amazonaws.jp/onca/xml?Service=AWSECommerceService
	UK	http://ecs.amazonaws.co.uk/onca/xml?Service=AWSECommerceService
	US	http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService
SOAP	CA	http://ecs.amazonaws.ca/onca/soap?Service=AWSECommerceService
	DE	http://ecs.amazonaws.de/onca/soap?Service=AWSECommerceService
	FR	http://ecs.amazonaws.fr/onca/soap?Service=AWSECommerceService
	JP	http://ecs.amazonaws.jp/onca/soap?Service=AWSECommerceService
	UK	http://ecs.amazonaws.co.uk/onca/soap?Service=AWSECommerceService
	US	http://ecs.amazonaws.com/onca/soap?Service=AWSECommerceService

NOTE: ECS 3.0 allowed you to use the *locale* parameter to indicate the Amazon store for which you wanted data. This parameter has been deprecated and there is no analogous option in ECS 4.0 – you simply indicate the locale by sending your request to the corresponding endpoint. For example, requests for Amazon.co.jp data should only go to the JP endpoint.

Request Parameters

ECS 4.0 has three types of request parameters:

- **Common parameters** are required or may be used in any ECS request. These parameters are used independently of the operation and response group.
- **Operation parameters** are required or optional parameters associated with specific operations.
- **Response group parameters** are generally never required, but may be used to control the behavior of one or more response groups. These parameters are relevant based on the response groups specified.

Each of these parameter types will be covered in greater detail throughout this guide.

Request Rate

The limitation to one request per second is enforced in ECS 4.0 just as in ECS 3.0.

Batch Requests and Multi-Operations

Batch requests and multi-operations allow you to combine multiple requests into a single call. Please see Appendix C for more information on these topics.

Common Parameters

This section covers the common parameters in ECS 3.0 requests and their counterparts in ECS 4.0.

Developer Token / Access Key Id

In ECS 3.0, each request must contain a valid Developer Token (passed as *dev*-t in REST and *devtag* in SOAP). ECS 4.0 takes an AWS Access Key Id (formerly known as SubscriptionId). Depending on when you received your token, it may not work with ECS 4.0. It is highly recommended that you obtain a new AWS Access Key Id from http://www.amazon.com/webservices. Please note that with ECS you will always use your Access Key Id and **never** your Secret Access Key.

ECS 3.0

```
http://xml.amazon.com/onca/xml3?
        dev-t=[Developer Token]
```

ECS 4.0

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]
```

Associate Id

The *AssociateTag* parameter in ECS 4.0 replaces the *t* (REST) and *tag* (SOAP) parameters in ECS 3.0. You must pass this parameter to receive referral commissions from the Amazon Associates program.

If you do not have an Associate ID or would like to create a new one, visit http://www.amazon.com/associates.

ECS 3.0:

```
http://xml.amazon.com/onca/xml3?
        dev-t=[dev token]&
        t=[associate id]
```

ECS 4.0:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        AssociateTag=[Associate ID]
```

Response Format

When using REST to access ECS 3.0, you are required to use the *f* parameter, which specifies either that an XML response is desired or it points to the location of an external XSLT to be applied to the response before it is returned. In ECS 3.0, even when XSLT is not being used, the *f* parameter is required to be set to “xml.”

In ECS 4.0, *f* is replaced by the *Style* parameter which can be used to specify the location of your XSLT file. If you are not using XSLT, this parameter is not required. If you need to set the MIME content type being returned in the response, you can set the *ContentType* parameter (which was the *ct* parameter in ECS 3.0). If no *ContentType* is set, the default is text/xml.

ECS 3.0:

```
http://xml.amazon.com/onca/xml3?
        dev-t=[dev token]&
        t=[associate id]&
        f=xml

http://xml.amazon.com/onca/xml3?
        dev-t=[dev token]&
        t=[associate id]&
        f=[XSLT URL]&
        ct=text/html
```

ECS 4.0:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        AssociateTag=[Associate Id]

http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        AssociateTag=[Associate Id]&
        Style=[XSLT URL]&
        ContentType=text/html
```

Response Type

Perhaps the largest difference between ECS 3.0 and 4.0 is the ability to specify what information is returned in the response. In ECS 3.0, there are only 2 options *lite* and *heavy*. However, ECS 4.0 gives you much finer control over the type of data returned. This is done using response groups. Response groups will be covered in greater depth later in this guide.

API Version

Another concept new to ECS 4.0 is API versioning. In ECS 3.0 there was no mechanism built into the API for future growth. This problem has been solved in ECS 4.0 by versioning each interface change made by Amazon. Versioning is done using dates where later dates indicate higher version numbers.

Versioning enables you to code against an interface that will not change. If you specify a particular WSDL version in your request, you are guaranteed that the response will match the types defined in the WSDL. When new functionality is added to ECS, it is only added to newer WSDL versions.

For example, the 2006-09-11 release introduced several enhancements to the offer data returned by ECS. Four new elements were added to the WSDL, one of which was the *AvailabilityType* field. In order to retrieve this field you would need to use a WSDL version of 2006-09-11 or later. Any requests with a prior WSDL version will not contain this data.

When making REST requests, the version is passed via the *Version* parameter. If *Version* is not specified, it will default to “2005-10-05”. It is strongly recommended you supply the *Version* parameter with every request.

When making SOAP requests, the version is not passed explicitly. Instead, it is passed as part of the SOAP envelope namespace. SOAP toolkits such as Axis and .NET take care of this automatically when you generate client code from the ECS WSDL. In order to upgrade WSDL versions, you must regenerate client code from the latest WSDL.

REST:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        AssociateTag=[Associate Id]&
        Version=2007-01-15
```

SOAP:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://ecs.amazonaws.com/AWSECommerceService/2007-01-15">
</soapenv:Envelope>
```

Operation

The *Operation* parameter specifies the particular API being requested. This parameter will be covered in much more detail in the "Operations" section.

Summary of Common Parameters

ECS 3.0 REST	ECS 3.0 SOAP	ECS 4.0
dev-t	devtag	AWSAccessKeyId
t	tag	AssociateTag (optional)
f	-	Style (optional, REST only)
ct	-	ContentType (optional, REST only)
type	type	ResponseGroup (optional)
-	-	Version (optional, REST only)
-	-	Operation

Operations

The concept of operations is very similar in ECS 3.0 and ECS 4.0. However, in ECS 4.0, you'll find that the number of operations has been dramatically reduced and that each operation is much more powerful than before.

Operations are also specified differently in ECS 4.0 than they were in ECS 3.0. In ECS 4.0, each request must specify the operation type using the *Operation* parameter. In addition, each operation typically has a set of other required or optional parameters affect the behavior of that operation. The table below shows each ECS 3.0 operation and its corresponding ECS 4.0 operation and operation parameters.

ECS 3.0	**ECS 4.0 Operation**	**Required Operation Parameters**
ActorSearch	ItemSearch	SearchIndex, Actor
ArtistSearch		SearchIndex, Artist
AuthorSearch		SearchIndex, Author
BlendedSearch		SearchIndex=Blended, Keywords
BrowseNodeSearch		SearchIndex, BrowseNode
DirectorSearch		SearchIndex, Director
KeywordSearch		SearchIndex, Keywords
ManufacturerSearch		SearchIndex, Manufacturer
PowerSearch		SearchIndex, Power
TextStreamSearch		SearchIndex, TextStream
AsinSearch	ItemLookup	ItemId
UpcSearch		IdType=UPC, ItemId, SearchIndex
ExchangeSearch	SellerListingLookup	SellerId, IdType=Exchange, Id
MarketplaceSearch=listing-id		SellerId, IdType=Listing, Id
MarketplaceSearch=keyword	SellerListingSearch	SellerId, Title
SellerSearch		SellerId
ListManiaSearch	ListLookup	ListType=Listmania, ListId
WishlistSearch		ListType=WishList, ListId
SellerProfile	SellerLookup	SellerId
SimilaritySearch	SimilarityLookup / ItemLookup*	ItemId
TransactionDetails	TransactionLookup	TransactionId
ShoppingCart=add (to create)	CartCreate	Items, Quantity
ShoppingCart=add (to add)	CartAdd	CartId, HMAC, Items, Quantity
ShoppingCart=modify	CartModify	CartId, HMAC, Items, Quantity, Action, CartItemId
ShoppingCart=remove		
ShoppingCart=clear	CartClear	CartId, HMAC
ShoppingCart=get	CartGet	CartId, HMAC

* Using a single ASIN, either the SimilarityLookup or ItemLookup operations can be used. See the section on "SimilarityLookup" for details.

The following sections address each ECS 4.0 operation and the ECS 3.0 operations that they replace. For each operation the important input parameters and response groups are covered. For a full listing of parameters and response groups, see the ECS 4.0 documentation.

ItemSearch

ECS 3.0 provides many different APIs for searching for products. The following ECS 3.0 operations have been consolidated into the ItemSearch operation:

- ActorSearch
- ArtistSearch
- AuthorSearch
- BlendedSearch
- BrowseNodeSearch
- DirectorSearch
- KeywordSearch
- ManufacturerSearch
- PowerSearch
- TextStreamSearch

Operation Parameters

ECS 3.0	ECS 4.0	Description
mode	SearchIndex	The search index to use; always required.
page	ItemPage	The page of results to retrieve.

The *mode* parameter from ECS 3.0 has been deprecated in favor of the *SearchIndex* parameter. A full list of values for *SearchIndex* is available in the ECS 4.0 technical documentation and in Appendix A. Unlike the *mode* parameter in ECS 3.0, *SearchIndex* values have been standardized across locales. As in ECS 3.0, this parameter is required for all searches.

ItemSearch has many other operation parameters that can be used to replicate the functionality delivered by the various ECS 3.0 searching operations. In addition, there are various enhancements that make searching with ECS 4.0 more powerful than ECS 3.0. See the ECS 4.0 documentation for further details.

Response Groups
ECS 3.0 operations return *ProductInfo* sets that contain Detail nodes for each item. An ItemSearch reqyest returns an *Items* element, which contains many *Item* elements. For more information on getting the corresponding response data, see the "Response Data" section.

ActorSearch

Use the ItemSearch operation with the *Actor* parameter:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemSearch&
        SearchIndex=DVD&
        Actor=[Actor Name]
```

ArtistSearch

Use the ItemSearch operation with the *Artist* parameter:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemSearch&
        SearchIndex=Music&
        Artist=[Artist Name]
```

AuthorSearch

Use the ItemSearch operation with the *Author* parameter:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemSearch&
        SearchIndex=Books&
        Author=[Author Name]
```

BlendedSearch

Use the ItemSearch operation with *SearchIndex=Blended* and the *Keywords* parameter:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemSearch&
        SearchIndex=Blended&
        Keywords=[Keywords]
```

For blended searches, *Keywords* is the only searching parameter that may be used.

BrowseNodeSearch

Use the ItemSearch operation with the *BrowseNode* parameter:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemSearch&
        SearchIndex=Electronics&
        BrowseNode=[Browse Node Id]
```

You can combine this with any of the other searching parameters such as *Keywords*, *Title*, *Author*, and/or *Manufacturer*:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemSearch&
        SearchIndex=Electronics&
        BrowseNode=[Browse Node Id]&
        Manufacturer=[Manufacturer Name]
```

NOTE: ECS 4.0 also has a BrowseNodeLookup operation for getting more data about browse nodes. See the ECS 4.0 documentation for more information.

DirectorSearch

Use the ItemSearch operation with the *Director* parameter:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemSearch&
        SearchIndex=DVD&
        Director=[Director Name]
```

KeywordSearch

Use the ItemSearch operation with the *Keywords* parameter:

http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
AWSAccessKeyId=*[Access Key Id]*&
Operation=ItemSearch&
SearchIndex=DVD&
Keywords=*[Keywords]*

ManufacturerSearch

Use the ItemSearch operation with the *Manufacturer* parameter:

http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
AWSAccessKeyId=*[Access Key Id]*&
Operation=ItemSearch&
SearchIndex=Tools&
Manufacturer=*[Manufacturer Name]*

PowerSearch

Use the ItemSearch operation with the *Power* parameter:

http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
AWSAccessKeyId=*[Access Key Id]*&
Operation=ItemSearch&
SearchIndex=Books&
Power=*[Power Search String]*

See the ECS 4.0 documentation to learn how to properly form a power search string.

TextStreamSearch

Use the ItemSearch operation with the *TextStream* parameter:

http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
AWSAccessKeyId=*[Access Key Id]*&
Operation=ItemSearch&
SearchIndex=Books&
TextStream=*[Text Stream String]*

NOTE: In addition to the searches shown here, you can now search multiple fields together. For example, you can search by Title, BrowseNode, and Actor:

http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
AWSAccessKeyId=*[Access Key Id]*&
Operation=ItemSearch&
SearchIndex=DVD&
Title=*[Title Words]*&
Actor=*[Actor Name]*&
BrowseNode=*[Browse Node Id]*

ItemLookup

ItemLookup replaces the following ECS 3.0 operations that enable you to find a product by a unique identifier, such as ASIN or UPC:

- AsinSearch
- UpcSearch

Operation Parameters

ECS 3.0	ECS 4.0	Description
-	IdType	Specifies what type of identifier is being used (default is "ASIN")
-	ItemId	The id of type specified by the *IdType* parameter. You can specify up to 10 item ids at a time.
mode	SearchIndex	Required when *IdType* is SKU, UPC, or EAN

Response Groups

Like ItemSearch, ItemLookup returns an *Items* element, which contains many *Item* elements. For more information on getting the corresponding response data, see the "Response Data" section.

AsinSearch

Use the ItemLookup operation. Because *ItemId* defaults to ASIN, it is optional for this request:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemLookup&
        ItemId=[ASIN]
```

UpcSearch

Use the ItemLookup operation with *IdType=UPC* and the *ItemId* and *SearchIndex* parameters. *SearchIndex* must be set just as *mode* needed to be set for UpcSearch:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemLookup&
        IdType=UPC&
        ItemId=[UPC]&
        SearchIndex=[Search Index]
```

NOTE: In addition to *IdType=ASIN* and *IdType=UPC* there are several other values that may be used in ECS 4.0. *See* the ECS 4.0 documentation for other possible values of *IdType*.

SellerListingLookup

The SellerListingLookup operation replaces the following ECS 3.0 APIs:

- ExchangeSearch
- MarketplaceSearch (using a listing-id)

Operation Parameters

ECS 3.0	ECS 4.0	Description
-	IdType	The type of identifier being used to lookup this listing.

		Either “Exchange” or “Listing”
▪ exchange_id (ExchangeSearch) ▪ listing_id (MarketplaceSearch)	Id	The exchange id or listing id of the listing
seller-id	SellerId	Required only when using a listing id

Response Groups
The default response group for SellerListingLookup is *SellerListing*. This response group will return all the same data as returned by ECS 3.0 with the exception of the seller rating. To obtain the value previously in the *ExchangeSellerRating* field you must use the SellerLookup operation separately or as part of a multi-operation.

ExchangeSearch

Use the SellerListingLookup operation with *IdType=Exchange* and *Id* parameter:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=SellerListingLookup&
        IdType=Exchange&
        Id=[Exchange Id]
```

MarketplaceSearch

MarketplaceSearches using a listing ID can be replaced using SellerListingLookup and *IdType=Listing*. The *SellerId* parameter (which was *seller-id* in ECS 3.0) must also be supplied.

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=SellerListingLookup&
        IdType=Listing&
        Id=[Listing Id]&
        SellerId=[Seller Id]
```

SellerListingSearch

The SellerListingSearch operation can be used to replace the following ECS 3.0 operations:

- MarketplaceSearch (for keywords)
- SellerSearch

Operation Parameters

ECS 3.0	ECS 4.0	Description
-	Title	Refine your search to only those listings containing matching title words
sort	Sort	Sorts the listings returned by a particular value. See the documentation for valid values.
page	ListingPage	Results are returned in pages of 10, this parameter indicates the page of listings to retrieve
offerstatus	OfferStatus	Filters results to either “Open” or “Closed” listings
seller_id	SellerId	Id of the seller whose listings you wish to search

Response Groups
The default response group for SellerListingSearch is *SellerListing*. This response group will return all the same data as ECS 3.0 with the exception of the seller rating. To obtain the value previously found in the

ExchangeSellerRating field you must use the SellerLookup operation separately or as part of a multi-operation.

MarketplaceSearch

MarketplaceSearches using keywords can be replaced using the SellerListingSearch operation with the *Title* parameter.

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=SellerListingSearch&
        SellerId=[Seller id] &
        Title=[Title keywords]
```

SellerSearch

Use the SellerListingSearch operation without the *Title* parameter to retrieve all a seller's listings:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=SellerListingSearch&
        SellerId=[Seller id]
```

ListLookup

The ListLookup operation in ECS 4.0 replaces the following ECS 3.0 operations:

- ListManiaSearch
- WishlistSearch

Operation Parameters

- *ListType*: The type of list to be retrieved.
- *ListId*: The id of the list to be retrieved.

Response Groups

The default response group for ListLookup is *ListInfo*. This response group provides basic information about the list including the name, date created, and number of items. To retrieve the particular items on a list, use the *ListFull* response group. You can then use any of the response groups described under the "Response Data" section to match the response data returned by *lite* and *heavy* in ECS 3.0.

NOTE: ECS 4.0 also has a ListSearch operation that enables you to search for Wish Lists, Wedding Registries, and Baby Registries by email address or name.

ListManiaSearch

Use the ListLookup operation with *ListType=Listmania*. You must specify the Listmania ID using the *ListId* parameter:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ListLookup&
        ListType=Listmania&
        ListId=[A listmania id]
```

WishlistSearch

Use the ListLookup operation with *ListType=WishList*. You must specify the Wish List ID using the *ListId* parameter:

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ListLookup&
        ListType=WishList&
        ListId=[A wish list id]
```

SellerLookup

The SellerLookup operation in ECS 4.0 replaces the already deprecated SellerProfile API in ECS 3.0.

Operation Parameters

ECS 3.0	ECS 4.0	Description
seller_id	SellerId	Id of the seller for which you are retrieving information

Response Groups

The only available (and default) response group for SellerLookup is *Seller*. This response group returns the seller's name, glance page URL, average feedback rating, and the last 10 reviews of the seller.

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=SellerLookup&
        SellerId=[Seller id]
```

SimilarityLookup

The SimilaritySearch API in ECS 3.0 can be replaced by either the SimilarityLookup operation or the ItemLookup operation depending on your usage.

One option is to use ItemLookup with the *Similarities* response group. This will return the ASIN and title for up to 5 similar products.

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemLookup&
        ItemId=[ASIN]&
        ResponseGroup=Similarities
```

However, the SimilarityLookup operation provides equivalent and enhanced behavior. SimilarityLookup takes between 1 and 10 ASINs as input and returns 10 *Item* nodes in the response. These 10 items can either be the set of products that are similar to **all** of the input ASINs or a random selection of items that are similar to one or possibly more of the input ASINs. When used with a single ASIN; however, 10 similar products will be returned just as they are by SimilaritySearch in ECS 3.0.

Operation Parameters

ECS 3.0	ECS 4.0	Description
-	ItemId	A list of up to 10 ASINs
-	SimilarityType	Indicates if you want to include only products that are similar to all of the items in the request (*Intersection*), or include an assortment of similar

		products corresponding to any of the items in the request (*Random*). The default value is *Intersection*.

Response Groups
SimilarityLookup returns a set of *Item* nodes likewise to the ItemSearch response. For more information on the response, see the "Response Data" section.

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=SimilarityLookup&
        ItemId=[ASIN]

http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=SimilarityLookup&
        ItemId=[ASIN 1],[ASIN 2]&
        SimilarityType=Intersection

http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=SimilarityLookup&
        ItemId=[ASIN 1],[ASIN 2]&
        SimilarityType=Random
```

TransactionLookup

The TransactionLookup operation in ECS 4.0 replaces the TransactionDetails API in ECS 3.0. This operation will allow you to retrieve certain information about an order with Amazon.

Operation Parameters

ECS 3.0	ECS 4.0	Description
OrderId	TransactionId	A list of up to 10 transaction/order ids

Response Groups
The only response group available with TransactionLookup is *TransactionDetails*.

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=TransactionLookup&
        TransactionId=[Transaction Id]
```

Cart Operations

Just as with ECS 3.0, ECS 4.0 enables you to create and manipulate remote shopping carts via the ECS web service. In ECS 3.0 there is a single ShoppingCart operation which takes five different types of actions:

- add
- modify
- remove
- get
- clear

The following ECS 4.0 operations replace the various cart operations in ECS 3.0:

- CartCreate
- CartAdd
- CartModify
- CartGet
- CartClear

Response Groups
With the exception of CartClear, all cart operations share a common set of possible response groups.

- *Cart*: The Cart response group returns the individual items in the cart, the subtotal, HMAC, *CartId*, and *PurchaseURL*. The *CartId* and HMAC must be used for follow-up cart requests. For convenience a URL encoded version of the HMAC is also provided to use for future REST requests. The *PurchaseURL* should be used to send your customers to the Amazon.com order pipeline to complete his or her order.

```
<Cart>
   <Request>…</Request>
   <CartId>[Cart Id]</CartId>
   <HMAC>[HMAC Value] </HMAC>
   <URLEncodedHMAC>[HMAC URL Encoded]</URLEncodedHMAC>
</Cart>
```

- *CartSimilarities*: Returns products that are similar to the ones in your cart.
- *CartTopSellers*: Returns the top five best sellers in the same category as the item used in the request (or a random item in your cart)
- *CartNewReleases*: Returns the top five new releases in the same category as the item used in the request (or a random item in your cart)

CartCreate

The CartCreate operation in ECS 4.0 replaces the *add* action in ECS 3.0 when you intend to create a new shopping cart. This operation creates a cart and adds the first item in a single request. You must specify at least one item to add to the new cart.

Operation Parameters

ECS 3.0	ECS 4.0	Description
Items	Items	Used for SOAP only; is a container element for Item elements
-	Item	Container for ASIN, OfferListingId, and Quantity elements. Used as a parameter prefix in REST (see below)
Asin	ASIN	The ASIN being added (see note below about adding by ASIN)
-	OfferListingId	Offer listing id of the offer to add to the cart
	Quantity	Quantity of the item to add to the cart

For each item being added to the cart, you must identify the item and specify the desired quantity. Items may be identified by either *ASIN* or *OfferListingId*.

NOTE: New applications using ECS 4.0 should always use *OfferListingId* as opposed to ASIN when adding items to a remote shopping cart. This parameter is more exact and will ensure you are buying from the correct merchant at the expected price. The ability to add an item by ASIN is only provided for backwards compatibility with existing ECS 4.0 applications.

Adding items to a cart is done differently depending on your request method. REST does not allow parameters to be nested as they can be in SOAP, therefore, the *Item* parameter is used to prefix each item parameter:

```
Item.X.ASIN
Item.X.OfferListingId
Item.X.Quantity
```

In the example, X is used as an index to associate each item identifier (ASIN or offer listing id) with the quantity. This index should start at 1 and increment sequentially.

REST

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=CartCreate&
        Item.1.OfferListingId=[Offer Listing Id]&
        Item.1.Quantity=[Quantity]&
        Item.2.OfferListingId=[Offer Listing Id]&
        Item.2.Quantity=[Quantity]
```

Using SOAP, each *Item* element can be nested inside the *Items* element.

SOAP

```
<soapenv:Envelope xmlns:soapenv=http://schemas.xmlsoap.org/soap/envelope/
xmlns:ns="http://ecs.amazonaws.com/AWSECommerceService/2007-01-15">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:CartCreate>
         <ns:AWSAccessKeyId>[Access Key Id]</ns:AWSAccessKeyId>
         <ns:Request>
            <ns:Items>
               <ns:Item>
                  <ns:OfferListingId>[Offer Listing Id]</ns:OfferListingId>
                  <ns:Quantity>[Quantity]</ns:Quantity>
               </ns:Item>
               <ns:Item>
                  <ns:OfferListingId>[Offer Listing Id]</ns:OfferListingId>
                  <ns:Quantity>[Quantity]</ns:Quantity>
               </ns:Item>
            </ns:Items>
         </ns:Request>
      </ns:CartCreate>
   </soapenv:Body>
</soapenv:Envelope>
```

CartAdd

CartAdd works the same way as CartCreate; however, you must supply the *CartId* and *HMAC* parameters. These values must come from a prior call to CartCreate.

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=CartAdd&
```

```
        CartId=[Cart Id from CartCreate]&
        HMAC=[HMAC from CartCreate]&
        Item.1.Quantity=[Quantity]&
        Item.1.OfferListingId=[Offer Listing Id]&
        Item.2.Quantity=[Quantity]&
        Item.2.OfferListingId=[Offer Listing Id]
```

CartModify

The CartModify operation can be used to remove an item from the cart, change the quantity of an item, or move items to and from the *Saved for Later* list. The *Saved for Later* list has items that are in the cart, but will not be purchased during checkout. In order to purchase these items, they must be moved back into the regular cart using CartModify.

To remove a single item from the cart, set the new quantity to 0

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=CartModify&
        CartId=[Cart Id]&
        HMAC=[HMAC]&
        Item.1.CartItemId=[Cart Item Id]&
        Item.1.Quantity=0
```

Change the quantity of an item in the cart

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=CartModify&
        CartId=[Cart Id]&
        HMAC=[HMAC]&
        Item.1.CartItemId=[Cart Item Id]&
        Item.1.Quantity=[New quantity]
```

Move an item from the cart into the “saved for later” items list

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=CartModify&
        CartId=[Cart Id]&
        HMAC=[HMAC]&
        Item.1.CartItemId=[Cart Item Id]&
        Item.1.Action=SaveForLater
```

Move an item from saved for later items back to the cart

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=04BNXW8MD74FWSQSX5R2&
        Operation=CartModify&
        CartId=[Cart Id]&
        HMAC=[HMAC]&
        Item.1.CartItemId=[Cart Item Id]&
        Item.1.Action=MoveToCart
```

CartClear

Use the CartClear operation to remove all items from a cart. Note this will only clear the active cart -- items that have been moved to *Saved for Later* will still remain.

http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
AWSAccessKeyId=*[Access Key Id]*&
Operation=CartClear&
CartId=*[Cart Id]*&
HMAC=*[HMAC]*

CartGet

Use the CartGet operation to retrieve the cart without performing any other action.

http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
AWSAccessKeyId=*[Access Key Id]*&
Operation=CartGet&
CartId=*[Cart Id]*&
HMAC=*[HMAC]*

Response Data

ECS 3.0 gave you very limited control over the data returned in responses. ECS 4.0 has solved this problem using *response groups*. Response groups enable you to specify exactly what type of data you want in the response. It is beneficial for you to limit the response groups used to smallest set possible. This will generate faster responses, utilize less bandwidth, and return less XML for your application to parse.

NOTE: *Large* and *Medium* provide a simplified interface for retrieving multiple response groups at once. They have been provided mainly for development and debugging purposes only. In your production applications you should specify the desired response groups individually as this will improve performance and response time.

A listing of all response groups is provided in Appendix B.

NOTE: The specific data returned by a response group may vary by WSDL version. Using an old WSDL version may limit the amount of data that is returned and the number of response groups that are available.

Item Data

Item data was previously controlled by the *heavy* and *lite* options in ECS 3.0. The following tables provide you with the equivalent response groups in ECS 4.0.

ECS 3.0	ECS 4.0
lite	Images
	ItemAttributes
	Offers
heavy	Accessories
	EditorialReview
	Images
	ItemAttributes
	Offers
	Reviews
	Similarities

Offer Data

ECS 3.0 allows you to use the “offer” parameter to set the type of offer data returned in the response. In ECS 4.0, this can be achieved using the *MerchantId* and *Condition* parameters in conjunction with any of the *Offers*, *OfferListings*, and *OfferFull* response groups (see documentation for the differences between the three). In addition, ECS 4.0 has the *OfferSummary* response group which contains summary information across all offers.

ECS 3.0 *offer*	ECS 4.0 *Condition*	ECS 4.0 *MerchantId*
-	New (default)	Amazon (default)
ThirdPartyNew	New (default)	All
used	Used	All
collectible	Collectible	All
refurbished	Refurbished	All
All	All	All

Variation Data

In ECS 3.0, the *Variation=true* parameter returns variation items when they exist. In ECS 4.0, there are several response groups dedicated to variations: *Variations*, *VariationSummary*, *VariationMinimum*, and *VariationImages*. Note that unless Amazon is selling the variations, you must set the *MerchantId* parameter.

Errors

Errors in ECS 4.0 have been designed to be much more informative than those in ECS 3.0. It is now much easier to diagnose problems with your request and determine what went wrong. Every error now contains an error code and a message describing the problem. Error codes are the same across ECS locales, however, the messages are now translated to the appropriate language. A full list of error codes and descriptions can be found in the ECS 4.0 documentation.

The following response fragment shows the error returned when an ItemSearch request is missing the *SearchIndex* parameter. Also shown is the *IsValid* element which contains a Boolean value indicating if the request was valid or not. Any request that causes this field to return False is invalid and will fail with an error.

```
<Items>
    <Request>
        <IsValid>False</IsValid>
        <Errors>
            <Error>
                <Code>AWS.MissingParameters</Code>
                <Message>
                    Your request is missing required parameters. Required parameters
                    include SearchIndex.
                </Message>
            </Error>
        </Errors>
    </Request>
</Items>
```

Error codes may appear at different levels within the XML response, depending on the operation and response groups used in your request.

- Requests missing the *Service* or *Operation* parameters will have the *Errors* node as the root element of the response

- Requests with a missing or invalid AWSAccessKeyId will have the *Errors* node under *OperationRequest* block

- Requests with missing or invalid operation or response group parameters will have the *Errors* node within the *Request* block of the response. There will be a *Request* block for each individual request (i.e. batch calls will have 2)

- In ECS 4.0 the placement of errors is the same for both REST and SOAP. Whereas ECS 3.0 contained errors within SOAP faults, ECS 4.0 does not.

Appendix A: Values for SearchIndex

The following list of search indices is accurate as of the 2007-01-15 release. Please see the latest ECS 4.0 documentation for the most recent version.

Search Index Name	US	UK	DE	JP	FR	CA
All	X					
Apparel	X					
Automotive	X					
Baby	X					
Beauty	X					
Blended	X	X	X	X	X	X
Books	X	X	X	X	X	X
Classical	X	X	X	X	X	X
DigitalMusic	X					
DVD	X	X	X	X	X	X
Electronics	X	X	X	X	X	
ForeignBooks			X	X	X	X
GourmetFood	X					
Grocery	X					
HealthPersonalCare	X	X	X	X		
Hobbies				X		
HomeGarden	X	X	X			
Industrial	X					
Jewelry	X					
Kitchen	X	X	X	X		
Magazines	X		X			
Merchants	X					
Miscellaneous	X					
Music	X	X	X	X	X	X
MusicalInstruments	X					
MusicTracks	X	X	X	X	X	
OfficeProducts	X					
OutdoorLiving	X	X	X			
PCHardware	X		X			
PetSupplies	X					
Photo	X		X			
Software	X	X	X	X	X	X
SoftwareVideoGames		X	X		X	X
SportingGoods	X		X	X		
Tools	X	X	X			
Toys	X	X	X	X		
VHS	X	X	X	X	X	X
Video	X	X	X	X		X
VideoGames	X	X	X	X	X	X
Wireless	X					
WirelessAccessories	X					

Appendix B: Response Groups by Operation

BrowseNodeLookup

- BrowseNodeInfo
- NewReleases
- Request
- TopSellers

CartAdd

- Cart
- CartNewReleases
- CartSimilarities
- CartTopSellers
- Request

CartClear

- Cart
- Request

CartCreate

- Cart
- CartNewReleases
- CartSimilarities
- CartTopSellers
- Request

CartGet

- Cart
- CartNewReleases
- CartSimilarities
- CartTopSellers
- Request

CartModify

- Cart
- CartNewReleases
- CartSimilarities
- CartTopSellers
- Request

CustomerContentLookup

- CustomerFull
- CustomerInfo
- CustomerLists
- CustomerReviews
- Request

CustomerContentSearch

- CustomerInfo
- Request

Help

- Help
- Request

ItemLookup

- Accessories
- AlternateVersions
- BrowseNodes
- Collections
- EditorialReview
- Images
- ItemAttributes
- ItemIds
- Large
- ListmaniaLists
- Medium
- MerchantItemAttributes
- OfferFull
- OfferListings
- Offers
- OfferSummary
- PromotionDetails
- PromotionSummary
- Request
- Reviews
- SalesRank
- Similarities
- Small
- Subjects
- Tracks
- VariationImages
- VariationMinimum
- Variations
- VariationSummary

ItemSearch

- Accessories
- AlternateVersions
- BrowseNodes
- Collections
- EditorialReview
- Images
- ItemAttributes
- ItemIds
- Large
- ListmaniaLists
- Medium

- MerchantItemAttributes
- OfferFull
- OfferListings
- Offers
- OfferSummary
- PromotionDetails
- PromotionSummary
- Request
- Reviews
- SalesRank
- SearchBins
- Similarities
- Small
- Subjects
- Tracks
- VariationMinimum
- Variations
- VariationSummary

ListLookup

- Accessories
- BrowseNodes
- EditorialReview
- Images
- ItemAttributes
- ItemIds
- Large
- ListFull
- ListInfo
- ListItems
- ListmaniaLists
- Medium
- OfferFull
- OfferListings
- Offers
- OfferSummary
- PromotionDetails
- PromotionSummary
- Request
- Reviews
- SalesRank
- Similarities
- Small
- Subjects
- Tracks
- VariationMinimum
- Variations
- VariationSummary

ListSearch

- ListInfo
- ListMinimum

- Request

SellerListingLookup

- Request
- SellerListing

SellerListingSearch

- Request
- SellerListing

SellerLookup

- Request
- Seller

SimilarityLookup

- Accessories
- BrowseNodes
- EditorialReview
- Images
- ItemAttributes
- ItemIds
- Large
- ListmaniaLists
- Medium
- OfferFull
- OfferListings
- Offers
- OfferSummary
- PromotionDetails
- PromotionSummary
- Request
- Reviews
- SalesRank
- Similarities
- Small
- Subjects
- Tracks
- VariationMinimum
- Variations
- VariationSummary

TransactionLookup

- Request
- TransactionDetails

Appendix C: Advanced Topics

Batch Requests

ECS 4.0 allows you to combine two requests of the same operation into a single request. The sample request below combines two different ItemSearch requests into a single REST request. The response will contain the data for each request separately.

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemSearch&
        ItemSearch.1.SearchIndex=[Search Index]&
        ItemSearch.2.SearchIndex=[Search Index]&
        ItemSearch.Shared.Keywords=[Keywords used in both searches]&
        ItemSearch.1.Author=[An author to be used in the first search]
```

SOAP requests can nest multiple *Request* nodes within the SOAP body and use the *Shared* block for parameters that are common between the two requests.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://ecs.amazonaws.com/AWSECommerceService/2007-01-15">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:ItemSearch>
         <ns:AWSAccessKeyId>[AWS Access Key Id]</ns:AWSAccessKeyId>
         <ns:Shared>
            <ns:Keywords>[Keywords used in both searches]</ns:Keywords>
         </ns:Shared>
         <ns:Request>
            <ns:SearchIndex>[Search Index]</ns:SearchIndex>
            <ns:Author>[An author to be used in the first search]</ns:Author>
         </ns:Request>
         <ns:Request>
            <ns:SearchIndex>[Search Index]</ns:SearchIndex>
         </ns:Request>
      </ns:ItemSearch>
   </soapenv:Body>
</soapenv:Envelope>
```

For more information on batch requests see the ECS Programming Guide in the technical documentation.

Multi-Operation Requests

In addition to batch requests, ECS 4.0 allows you to submit two operations in one request. The sample request below combines an ItemSearch and an ItemLookup into a single REST request. Similarly to batch requests, the response will contain separate sections of data for each request.

```
http://ecs.amazonaws.com/onca/xml?Service=AWSECommerceService&
        AWSAccessKeyId=[Access Key Id]&
        Operation=ItemSearch,ItemLookup&
        ItemSearch.1.SearchIndex=[Search Index]&
        ItemSearch.1.Keywords=[Search Keywords]&
        ItemLookup.1.ItemId=[ASIN]
```

SOAP requests can nest multiple operation blocks with the main SOAP body.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://ecs.amazonaws.com/AWSECommerceService/2007-01-15">
    <soapenv:Header/>
    <soapenv:Body>
        <ns:MultiOperation>
            <ns:ItemSearch>
                <ns:AWSAccessKeyId>[AWS Access Key Id]</ns:AWSAccessKeyId>
                <ns:Request>
                    <ns:Keywords>[Search keywords]</ns:Keywords>
                    <ns:SearchIndex>[Search Index]</ns:SearchIndex>
                </ns:Request>
            </ns:ItemSearch>
            <ns:ItemLookup>
                <ns:AWSAccessKeyId>[AWS Access Key Id]</ns:AWSAccessKeyId>
                <ns:Request>
                    <ns:ItemId>[ASIN]</ns:ItemId>
                </ns:Request>
            </ns:ItemLookup>
        </ns:MultiOperation>
    </soapenv:Body>
</soapenv:Envelope>
```

For more information on multi-operations, see the ECS Programming Guide in the technical documentation.