The Callback API Guide

A reference guide for developers who want to calculate their own shipping or promotional discounts for their Checkout by Amazon orders.
Table of Contents

1 Overview of the Callback API ........................................................................................................1
   1.1 Prerequisites for Using the Callback API ......................................................................................1
   1.2 Your Tasks When Using the Callback API ........................................................................................2
   1.3 Other Documentation .....................................................................................................................2
   1.4 What’s New in This Document .......................................................................................................2
   1.5 Sample Code ................................................................................................................................3

2 Enable Callbacks in Your Cart XML ...............................................................................................4
   2.1 Enabling Callbacks Example ............................................................................................................4

3 Handle Callback Requests from Amazon Payments .............................................................................6
   3.1 The Callback Content ......................................................................................................................6
      3.1.1 1. The Signature .........................................................................................................................7
      3.1.2 2. The UUID ..............................................................................................................................7
      3.1.3 3. The Timestamp .......................................................................................................................7
      3.1.4 4. The Order Calculation Request .........................................................................................8
   3.2 Processing the Callback Request ..................................................................................................9
      3.2.1 1. Parse the Request Data ........................................................................................................10
      3.2.2 2. Verify That the Callback Request Is Valid ...........................................................................10
      3.2.3 3. Calculate Promotional Discounts or Shipping Rates ............................................................10
      3.2.4 4. Prepare the Response to Amazon Payments ........................................................................10

4 Return Callback Response to Amazon Payments ............................................................................11
   4.1.1 1. The Order Calculation Response ...............................................................................................12
   4.1.2 2. The Signature ..........................................................................................................................15
   4.1.3 3. The Seller Central Access Key ID ............................................................................................15
   4.1.4 The Promotions Response ..........................................................................................................15
   4.1.5 The ShippingMethods Response ...............................................................................................16
   4.2 More Examples of Callback Responses ...........................................................................................17
      4.2.1 Shipping Using All Shipping Speeds ...........................................................................................17
      4.2.2 Excluding a Shipping Speed ......................................................................................................18
      4.2.3 Excluding an Address Entirely ...................................................................................................19
   4.3 Adding an Error Response ..............................................................................................................19
      4.3.1 Error Response Example ..........................................................................................................20
      4.3.2 Error Response Definitions ....................................................................................................20
   4.4 How the ProcessOrderOnCallbackFailure Tag Is Used ...............................................................21
   4.5 Order Processing on Fallback Failure Table ....................................................................................23
   4.6 Error Messages the Buyer Sees ....................................................................................................24
   4.7 Error Responses You Receive .......................................................................................................24
      4.7.1 Error Codes and Messages .......................................................................................................25

5 Using Custom Data Fields with Callbacks .....................................................................................28
   5.1 Prerequisites for Using Custom Data Fields ....................................................................................28
   5.2 How We Process the Custom Data Fields .......................................................................................28
   5.3 Modifying Your Order XML to Use Custom Data Fields ...............................................................28
   5.4 Using Custom Data Fields with the Callback API .........................................................................29
   5.5 Using a Custom Namespace as Part of the Custom Data Entry ......................................................30

6 Using Real-time Cart Updates with the Callback API ......................................................................32
   6.1 How Real-time Cart Updates Works ...............................................................................................32
1 Overview of the Callback API

The Callback API lets you specify shipping and promotional discount amounts using your own application logic at the time an order is placed when using Checkout by Amazon.

This illustration shows the sequence of events.

1.1 Prerequisites for Using the Callback API

Before you can use the Callback API, your systems must meet the following prerequisites:

- You must create your cart using XML. The Callback API does not currently support carts created using HTML nor does it support carts created using the Generate Button feature in Seller Central.
- You must be using the order.xsd with a file date of 2009-05-15 or later. (Download this version here.)
- You must be using the callback.xsd with a file date of 2009-05-15 or later. (Download this version here.)
- Each Item in your cart XML must include the SKU element.
You must build and operate a fast, reliable webservice to calculate shipping and promotional discounts based on the buyer’s address and other information we send you.

You must respond to the callback request from Amazon Payments within five seconds of us sending the message to your endpoint URL.

You must accept security credentials from any of the sources listed in Appendix C: Accepted SSL Certificates.

1.2 Your Tasks When Using the Callback API

You have three tasks to accomplish when using the Callback API:

1. **Enable Callbacks** – You enable callbacks by adding the `OrderCalculationCallbacks` section to your original cart XML.
2. **Handle Callbacks** – You operate a fast, reliable webservice that accepts our requests and calculates the correct shipping and promotional discounts for the order.
3. **Return Callback Requests** – You return the information you calculated to us.

The rest of this document discusses how to accomplish these three tasks.

1.3 Other Documentation

You can read our downloadable files (PDF) to understand more about Checkout by Amazon.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Audience, Purpose, and Goals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Getting Started Guide (PDF)</td>
<td>For merchants who want to use a simple, step-by-step approach in setting up Checkout by Amazon on their websites, from start to finish. The Getting Started Guide walks merchants through the process, from gathering the required information for signing up for an account setting up a button on their websites.</td>
</tr>
<tr>
<td>Implementation Guide Using the Create Checkout Button Feature (PDF)</td>
<td>For merchants who want detailed information about integrating their website with Checkout by Amazon using the Create Checkout Button feature in Seller Central. Designed for a simple copy-and-paste HTML solution for your webpages.</td>
</tr>
<tr>
<td>Implementation Guide Using HTML-based Button Code and Shopping Carts (PDF)</td>
<td>For merchants and developers who want detailed information about integrating their website with Checkout by Amazon using HTML-based buttons and shopping carts. Designed for HTML-based integration.</td>
</tr>
<tr>
<td>Implementation Guide Using XML-based Button Code and Shopping Carts (PDF)</td>
<td>For merchants and developers who want detailed information about integrating their website with Checkout by Amazon using XML-based buttons and shopping carts. Designed for XML-based integration.</td>
</tr>
<tr>
<td>Using Seller Central (PDF)</td>
<td>For merchants who want an overview of Seller Central.</td>
</tr>
<tr>
<td>Best Practices Guide (PDF)</td>
<td>Contains the best tips for merchants who want to set up their websites to use Amazon Payments, with advice for planning, setting up, and day-to-day operations.</td>
</tr>
<tr>
<td>Launch Checklist (PDF)</td>
<td>For merchants and developers planning to set up and use Checkout by Amazon on their site.</td>
</tr>
</tbody>
</table>

1.4 What’s New in This Document

<table>
<thead>
<tr>
<th>Ver</th>
<th>Date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>2011-03-15</td>
<td>Initial Release</td>
</tr>
</tbody>
</table>
1.5 Sample Code

We have Java sample code (v 2009-08-25.1.0.2, using schema date 2009-05-15) and PHP sample code (v 2014-03-05.1.0.2) you can review to learn how to implement the Callback API in your cart XML.
2 Enable Callbacks in Your Cart XML

To use the Callback API, you include a new `OrderCalculationCallbacks` section in your cart XML, within the `<Order>...</Order>` section and below the `<Cart>...</Cart>` section, with information instructing us which aspects of the order you will calculate yourself: promotional discounts, shipping costs, or both.

2.1 Enabling Callbacks Example

Here is an example of the XML portion of an order with the `OrderCalculationCallbacks` section, specifying that you will calculate the shipping and promotional discounts for the order when you receive our callback request.

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns="Checkout by Amazon Shopping Cart">
  <Cart>
    <Items>
      <Item>
        <SKU>JKL909</SKU>
        <MerchantId>AEIOU1234AEIOU</MerchantId>
        <Title>Calvin and Hobbes Reliquary</Title>
        <Price>
          <Amount>29.99</Amount>
          <CurrencyCode>GBP</CurrencyCode>
        </Price>
        <Quantity>1</Quantity>
      </Item>
    </Items>
  </Cart>
  <!-- THIS IS THE NEW SECTION YOU ADD TO ENABLE CALLBACKS -->
  <OrderCalculationCallbacks>
    <CalculatePromotions>true</CalculatePromotions>
    <CalculateShippingRates>true</CalculateShippingRates>
    <OrderCallbackEndpoint>https://my.endpoint.com/receive.php</OrderCallbackEndpoint>
    <ProcessOrderOnCallbackFailure>true</ProcessOrderOnCallbackFailure>
  </OrderCalculationCallbacks>
</Order>
```

Notes

References surrounded by /*...*/ are comments for this guide only; you will not see them in code responses you receive from us, and you do not need to include them in the cart XML you send to Checkout by Amazon.
This example is the XML portion of the order; see “Base64-encode the Cart Data” Implementation Guides, found in the Technical Resources help section, for information on how to prepare your cart XML.

Be sure you specify an endpoint using https:// for your Production code and that you include a SKU for every Item.

The sequence of tags in your cart XML and your response XML must match the order listed in the order.xsd and the callback.xsd; otherwise, the request and response are rejected.
3 Handle Callback Requests from Amazon Payments

Checkout by Amazon sends a callback request to your endpoint URL using the POST command each time any of the following events occur:

- The buyer selects or changes a shipping address while in the order pipeline.
- The buyer reaches the Place Your Order page after clicking a Checkout button (bypassing any page to select a shipping address).

For every callback, your specified endpoint URL receives a request which your webservice must process and promptly respond to within the required time (five seconds). The content of the request is defined in the `callback.xsd` document. See Appendix B: The Callback Schema for more information.

Notes

For a Production order, we send the callback request to you over HTTPS, using port 443. For a Sandbox order, we send the callback request to you over HTTP, using port 80.

You will not receive a callback request if your cart XML does not include the correct namespace, if it is missing a SKU for any Item, or if you do not include an endpoint for Production orders specified using the https:// protocol.

Do not assume that a callback request means you should reserve inventory for the order. The order is not placed until the checkout process is complete and you get an order notification, whether as a report or within Seller Central.

You must accept security credentials from any of the sources listed in Appendix C: Accepted SSL Certificates.

3.1 The Callback Content

The callback request is a simple string, URL-encoded and formatted using UTF-8. Depending on your signed-cart settings in Seller Central, the request can contain either one or four of the following four sections.

<table>
<thead>
<tr>
<th>Entry</th>
<th>Definition</th>
<th>Inclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The <strong>Signature</strong></td>
<td>Calculated using the UUID, the timestamp, and your Public Key</td>
<td>Only for signed carts</td>
</tr>
<tr>
<td>2. The <strong>UUID</strong></td>
<td>A Universally Unique Identifier</td>
<td>Only for signed carts</td>
</tr>
<tr>
<td>3. The <strong>Timestamp</strong></td>
<td>The request time using GMT</td>
<td>Only for signed carts</td>
</tr>
<tr>
<td>4. The <strong>order calculation request</strong></td>
<td>The content of the request</td>
<td>Always included</td>
</tr>
</tbody>
</table>

Note that Signature and Timestamp have an initial capital letter; UUID is all capital letters, and order-calculations-request is all lowercase letters.

This request is a set of key-value pairs, concatenated as one string, with each key-value pair delimited using the standard “&” character. The key is in plain text, and the value is URL-encoded. If you did
not specify in Seller Central that you accept only signed carts, then the string contains only the order calculation request. To review or change your signed-carts setting, go to Seller Central, click the Settings tab, and then click Checkout Pipeline Settings.

Note that the request contains the following key-value pairs. The order is not important:

Signature=[Signature_value] & UUID=[UUID_value] & Timestamp=[Timestamp_value] & order-calculations-request=[request_value]

The next few pages discuss the four components the callback request you receive, and how you can use them to build your callback response.

3.1.1 1. The Signature
To create the signature, we concatenate the UUID and Timestamp, calculate an HMAC SHA-1 signature using your Seller Central Access Key ID, and then URL encode it.

Here is an example of the signature in the request you receive.

Signature=MRYUB4bk%2B3ehw0BAbk6mp6SxXw%3D

You'll use this Signature later when you calculate the signature using your Seller Central Secret Key to validate the request. (If you did not specify that you accept only signed carts, you will not receive the Signature in your merchant request.)

3.1.2 2. The UUID
The UUID (Universally Unique Identifier) is a string created by AWS (Amazon Webservices).

Here is an example of the UUID in the request you receive:

UUID=d6a58609-d9ea-415c-95c6-d7c2528fca09

You'll use this UUID later when you calculate the signature to validate the request. (If you did not specify that you accept only signed carts, you will not receive the UUID in your merchant request.)

3.1.3 3. The Timestamp
The timestamp is the date and time the request was sent from our servers.

Here is an example of the timestamp in the request you receive:

Timestamp=2008-11-14T20%3A57%3A07.146Z

You’ll use this Timestamp later when you calculate the signature to validate the request. (If you did not specify that you accept only signed carts, you will not receive the Timestamp in your merchant request.)

3.1.4 4. The Order Calculation Request

Here is an example of the order calculation request expressed as readable XML.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OrderCalculationsRequest xmlns="Checkout by Amazon Shopping Cart">
/*               */
/* THIS IS THE UNIQUE CALLBACKREFERENCEID FOR THIS REQUEST */
/*               */
<CallbackReferenceId>1-f1b06763-b30c-47de-ab95-b4d8fef91ef5</CallbackReferenceId>
/*               */
/* THIS SECTION IS A COPY OF THE MERCHANTCALCULATIONCALLBACK */
/* SECTION YOU SENT IN THE CART XML */
/*               */
<OrderCalculationCallbacks>
  <CalculatePromotions>true</CalculatePromotions>
  <CalculateShippingRates>true</CalculateShippingRates>
  <OrderCallbackEndpoint>https://my.endpoint.com/receive.php</OrderCallbackEndpoint>
  <ProcessOrderOnCallbackFailure>true</ProcessOrderOnCallbackFailure>
</OrderCalculationCallbacks>
/*               */
/* THIS SECTION IS A COPY OF THE ORDER INFORMATION IN THE CART XML */
/*               */
<ClientRequestId>1A2B3C4D5E</ClientRequestId>
<IntegratorId>VWXYZ98765VWXYZ</IntegratorId>
<IntegratorName>IntegratorWorld</IntegratorName>
<Cart>
  <Items>
    <Item>
      <SKU>JKL909</SKU>
      <MerchantId>AEIOU1234AEIOU</MerchantId>
      <Title>Calvin and Hobbes Reliquary</Title>
      <Description>By Bill Watterson</Description>
      <Price>
        <Amount>29.99</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </Price>
      <Quantity>3</Quantity>
      <Weight>
        <Amount>2.0</Amount>
        <Unit>kg</Unit>
      </Weight>
      <URL>http://mysite.com/item?909</URL>
      <Category>Books</Category>
      <Condition>New</Condition>
      <FulfillmentNetwork>MERCHANT</FulfillmentNetwork>
      <Images>
        <Image>
          <URL>http://mysite.com/pix?alpha.jpg</URL>
        </Image>
      </Images>
    </Item>
  </Items>
</Cart>
/*               */
/* THIS SECTION IS THE SHIPPING INFORMATION FOR THE ORDER */
```
The request contains the CallbackReferenceId (a unique ID generated for each callback request), the OrderCalculationCallbacks section (copied from your cart XML), the ClientRequestId, IntegratorId, and IntegratorName (copied from your cartXML), the cart (copied from your cart XML), and CallbackOrders, a new section representing the information your service will use to calculate the shipping or promotional discounts for the order. This section contains the buyer’s address as well as the SKU that will be shipped to the address. Your webservice must take this information, calculate the shipping or promotional discounts, and then return the results to Checkout by Amazon.

3.2 Processing the Callback Request

You use the information in the callback request (the XML content as well as the other information in the request) to perform the following actions:

1. Parse the request data.
2. Verify that the callback request is valid (compare signatures, if you specified that you accept only signed carts).
3. Calculate the shipping and promotional discounts.
4. Prepare the response to Amazon Payments.

Your application must parse this request to verify that the callback is genuinely from Amazon Payments, process the information for the merchant calculation, and prepare a response.
3.2.1 1. Parse the Request Data
Your first step is to parse the request data. Note that the ampersand “&” character separates the values, and that request is URL-encoded.

3.2.2 2. Verify That the Callback Request Is Valid
The next step is to compare the signature we generate with the signature you generate to verify that the response is genuine so you do not respond to requests that might incorrectly claim to be from Amazon Payments.

Note
You can skip this step if you did not specify in Seller Central that you accept only signed carts.

To calculate the signature, you decode the URL-encoded UUID and Timestamp from the original request; concatenate the UUID and Timestamp; then use a standard HMAC SHA1 algorithm and your Seller Central Secret Access Key (your private key). See Manage Your Security Keys in Seller Central online help for information on generating and viewing your Seller Central Access Key ID and your Seller Central Secret Access Key.

Notes
You must use your Seller Central Secret Key.

Be sure you use the Seller Central Access Key ID and Seller Central Secret Access Key to validate and generate signatures for requests and responses.

After you verify the signature, you can proceed with processing the XML document.

3.2.3 3. Calculate Promotional Discounts or Shipping Rates
After parsing the information in the request we send you, your application must calculate the amounts for promotional discounts or shipping using the buyer’s shipping address.

3.2.4 4. Prepare the Response to Amazon Payments
To prepare the response to our request, you will assemble the results into an XML document that follows the schema in the callback.xsd.

Your service should automatically respond to our callback request. The next few pages discuss how to do this.

Note
Just as you received the request as a URL-encoded string, your response must also be URL-encoded.
4 Return Callback Response to Amazon Payments

Your callback response must be URL-encoded. However, your response will contain different key-value pairs. Depending on your signed-cart settings in Seller Central, your response can contain either one or three of the following three sections.

<table>
<thead>
<tr>
<th>Entry</th>
<th>Definition</th>
<th>Inclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The <strong>order calculation response</strong></td>
<td>The content of the request</td>
<td>Always included</td>
</tr>
<tr>
<td>2. The <strong>Signature</strong></td>
<td>Calculated using the content (response XML) and your Seller Central Secret Access Key</td>
<td>Only for signed carts</td>
</tr>
<tr>
<td>3. The Seller Central <strong>Access Key ID</strong></td>
<td>Your Seller Central Access Key ID</td>
<td>Only for signed carts</td>
</tr>
</tbody>
</table>

**Notes**

Signature has an initial capital letter; `order-calculations-response` and `aws-access-key-id` are all lowercase letters.

Do not send your Seller Central Secret Access Key – this is a value only you should know and that you should keep secure.

The length of the callback URL must not be longer than 1000 characters. Longer callback responses are truncated at the 1000th character.

Your response is a set of key-value pairs, concatenated as one string and encoded using URL encoding, with each key-value pair delimited using the standard “&” character. The key is in plain text, and the value is URL-encoded. If you did not specify in Seller Central that you accept only signed carts, then the string contains only the `order-calculations-response`. To review or change your signed-carts setting, go to Seller Central, click the Settings tab, and then click Checkout Pipeline Settings.

Note that the response is as follows. The order is not important:

```
order-calculations-response=[response_value] & Signature=[Signature_value] & aws-access-key-id=[access_key_value]
```

Your callback response must contain responses for every element requested and only for the element requested; that is, if the original cart XML specified only `CalculateShippingRates` as “true”, then your response must include the calculated shipping rate amount and no other calculated amounts such as promotional discounts.

If your callback response contains more information or less information than requested, or the information is malformed, then the order processing continues based on the value of the `ProcessOrderOnCallbackFailure` flag. See How the ProcessOrderOnCallbackFailure tag Is Used for more information about how this flag is used.
Note

Before you return your response, we recommend you validate it with an XML parser to isolate schema issues.

4.1.1 1. The Order Calculation Response

This table lists the elements in your response:

<table>
<thead>
<tr>
<th>Entry</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response</td>
<td>Contains one of the two types of responses, CallbackOrders or Error.</td>
</tr>
<tr>
<td>CallbackOrders</td>
<td>Used to specify all the information for various CallbackOrders in the callback request. CallbackOrders must have at least one CallbackOrder section.</td>
</tr>
<tr>
<td>Promotions</td>
<td>Referenced from the order.xsd; contains the same information as Promotions element if it is present in the cart XML. Promotions contains definitions for promotion IDs specified at the CallbackOrderItem level for each CallbackOrderItem if requested.</td>
</tr>
<tr>
<td>ShippingMethods</td>
<td>Referenced from the order.xsd; contains the same information as ShippingMethods element if it is present in the cart XML. ShippingMethods contains definitions for ShippingMethodIds that are specified at the CallbackOrderItem level for each CallbackOrderItem if requested.</td>
</tr>
<tr>
<td>CartPromotionId</td>
<td>Referenced from the order.xsd; contains the same information as in the order.xsd. CartPromotionId is a string that represents the PromotionId that overrides all promotions on the order. If specified, this overrides any PromotionId specified at CallbackOrderItem level</td>
</tr>
<tr>
<td>Error</td>
<td>Indicates that an error occurred generating response in your webservice.</td>
</tr>
</tbody>
</table>

Valid Callback Response Example

Here is a valid callback response, formatted as readable XML, with promotions and shipping methods included in the response. It assumes an order going to a Seattle address, and provides four shipping options for the buyer. It also calculates the shipping charges, and provides a $10.00-off promotion.

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <Response>
    <CallbackOrders>
      <CallbackOrder>
        <Address>
          <AddressId>AddressId</AddressId>
        </Address>
      </CallbackOrder>
    </CallbackOrders>
    <CallbackOrderItems>
      <CallbackOrderItem>
        <SKU>JKL909</SKU>
        <PromotionIds>
          <PromotionId>TenPound</PromotionId>
        </PromotionIds>
        <ShippingMethodIds>
          <ShippingMethodId>US Standard</ShippingMethodId>
        </ShippingMethodIds>
      </CallbackOrderItem>
    </CallbackOrderItems>
  </Response>
</OrderCalculationsResponse>
```
<Promotions>
  <Promotion>
    <PromotionId>TenPound</PromotionId>
    <Description>Ten Pounds Off</Description>
    <Benefit>
      <FixedAmountDiscount>
        <Amount>10.0</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </FixedAmountDiscount>
    </Benefit>
  </Promotion>
</Promotions>

<ShippingMethods>
  <ShippingMethod>
    <ShippingMethodId>US One-day</ShippingMethodId>
    <ServiceLevel>OneDay</ServiceLevel>
    <Rate>
      <ItemQuantityBased>
        <Amount>10.29</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </ItemQuantityBased>
    </Rate>
    <IncludedRegions>
      <USZipRegion>98104</USZipRegion>
    </IncludedRegions>
  </ShippingMethod>
  <ShippingMethod>
    <ShippingMethodId>US Standard</ShippingMethodId>
    <ServiceLevel>Standard</ServiceLevel>
    <Rate>
      <WeightBased>
        <Amount>3.49</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </WeightBased>
    </Rate>
    <IncludedRegions>
      <USZipRegion>98104</USZipRegion>
    </IncludedRegions>
  </ShippingMethod>
  <ShippingMethod>
    <ShippingMethodId>US Two-day</ShippingMethodId>
    <ServiceLevel>TwoDay</ServiceLevel>
    <Rate>
      <ShipmentBased>
        <Amount>15.49</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </ShipmentBased>
    </Rate>
  </ShippingMethod>
</ShippingMethods>
Notes

The sequence of tags in your cart XML and your response XML must match the order listed in the order.xsd and the callback.xsd definitions; otherwise, the request and response are rejected by our XML parser. See Appendix B: The Callback Schema for more information about the callback.xsd.

You use your own applications to determine the rates you want to be used for shipping, as well as any promotional discounts, based on the information we send you in our callback request. When we receive your callback response, we use your information to calculate shipping (your shipping charge type) and promotional discounts (your type of promotion and the amount of the discount along with the information you specified in the original cart XML, such as the number of items and their weight.

If you are using ItemBased shipping, then the Rate you send back should assume a per-unit charge (each Item, per pound, for example).

When you set the IncludedRegions, you can just tell us to use ZIP Code (USZip/ZIP Code) or PredefinedRegion/WorldRegion. We need the other information – the shipping rate or promotional discount to use as well as the currency code – to figure out the calculations for the order.

If you specify PromotionCalculations or ShippingCalculationRates in your OrderCalculationCallbacks section, we ignore the values specified elsewhere in the cart XML or in Seller Central. If you do not specify PromotionCalculations or ShippingCalculationRates in your OrderCalculationCallbacks section, or your response for these value fails (for example, it's not formatted correctly), then we look at the values (if any) you supply in the Order section of the cart XML or in Seller Central.
4.1.2 2. The Signature
To create the signature, concatenate the XML, calculate an HMAC SHA-1 signature using your Seller Central Secret Access Key (your private key), and then URL encode the result. See Manage Your Security Keys in Seller Central online help for information on generating and viewing your Seller Central Access Key ID and your Seller Central Secret Access Key.

**Notes**

You can skip this step if you did not specify in Seller Central that you accept only signed carts.

You must use your Seller Central Access Key ID or Seller Central Secret Access Key.

Be sure you use the Seller Central Access Key ID and Secret Access Key to validate or generate signatures for requests and responses.

Your Signature entry will look something like this:

```
Signature=A1c%5Cegh%2839sjjOpbeadd730%21+aao032%3D
```

4.1.3 3. The Seller Central Access Key ID
The final step is to URL-encode and then attach your Seller Central Access Key ID.

**Notes**

You can skip this step if you did not specify in Seller Central that you accept only signed carts.

Be sure you attach your Seller Central Access Key ID and not your Seller Central Secret Access Key. The Secret Access Key is the private key known only to you.

Your Seller Central Access Key entry (aws-access-key-id) will look something like this:

```
aws-access key-id=QVJ3DCYP1HJ29X7Y802
```

4.1.4 The Promotions Response.
The behavior is similar to the promotions tables (overrides) you can set up in your cart XML. You assign one PromotionId value to a CallbackOrderItem. Then, you build a Promotions section with a promotion Benefit that applies to a CallbackOrderItem in your order. You can have multiple Promotion entries in your response, each with its own PromotionId and Benefit values.
Notes

Promotional benefits for overrides and Seller Central are *additive*. If you set up a valid promotion in your callback response, then only that promotion is used – any other promotion specified in the cart XML or in Seller Central is ignored. But if your callback response promotion fails and you've set the ProcessOrderOnCallbackFailure flag to “true”, then any override promotions or in Seller Central are evaluated and then used if applicable. This means it's possible that a benefit in your override *and* in Seller Central *both* get applied to the order.

You can specify promotional benefits either at the cart level (“Get $5.00 off your entire order”) or at the item level (“Get $5.00 off each widget in your order.”). However, you can't have both cart-level and item-level promotions in the same response. If you include both, then only the cart-level promotion is used.

You can have only one item-level promotion per item. However, if the order contains multiple items, each item can have its own promotion.

4.1.5 The ShippingMethods Response

Your ShippingMethods entry is similar to the shipping tables (overrides) you can set up in your cart XML.

First, you build a ShippingMethodIds in your CallbackOrderItems section with multiple ShippingMethodId entries that relate to the CallbackOrderMethod values you will support.

Then, you build a ShippingMethods section and specify the CallbackOrder service levels, rates, and supported region.

Your callback response can assign multiple ShippingMethodId entries for each CallbackOrderItem. It can also include multiple ShippingMethods defined with different ServiceLevel entries. Note that you can have only one ShippingMethod for each ServiceLevel. You must always specify an IncludedRegions entry for a ShippingMethod. You can optionally specify the ExcludedRegions entry as well.

If you determine that you ship to the buyer's address, simply use `<USZipRegion>(buyer’s ZIP Code)</USZipRegion>` (for U.S.-based orders) or `<PredefinedRegion>WorldAll</PredefinedRegion>` (for International orders) as your IncludedRegions. If you determine that you do not ship to the buyer’s address, use `<USZipRegion>(buyer’s ZIP Code)</USZipRegion>` (for U.S.-based orders) or `<PredefinedRegion>WorldAll</PredefinedRegion>` for both your IncludedRegions and your ExcludedRegions.
Note

Shipping rate assignments for overrides and Seller Central are exclusive. If you set up a valid shipping rate in your callback response, then only that shipping rate is used – any other shipping rates specified in the cart XML or in Seller Central are ignored. If you don’t include valid shipping information in your callback response and you’ve set `CalculateShippingRates` to “true”, then we use specified shipping overrides in the cart XML for shipping rate information when we process the order. If you do not specify shipping overrides in the cart XML, then we use the shipping settings in Seller Central.

4.2 More Examples of Callback Responses

Here are a few more code samples to review to understand constructing your callback response.

4.2.1 Shipping Using All Shipping Speeds

In this example, all four shipping speeds are supported. To specify the supported shipping service levels, just include the `IncludedRegions` value set to `USZipRegion` matching the buyer’s ZIP Code.

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <Response>
    <CallbackOrders>
      <CallbackOrder>
        <Address>
          <AddressId>12345</AddressId>
        </Address>
        <CallbackOrderItems>
          <CallbackOrderItem>
            <SKU>JKL909</SKU>
            <ShippingMethodIds>
              <ShippingMethodId>US Standard</ShippingMethodId>
              <ShippingMethodId>US Expedited</ShippingMethodId>
              <ShippingMethodId>US One-day</ShippingMethodId>
              <ShippingMethodId>US Two-day</ShippingMethodId>
            </ShippingMethodIds>
          </CallbackOrderItem>
        </CallbackOrderItems>
      </CallbackOrder>
    </CallbackOrders>
  </Response>
  <ShippingMethods>
    <ShippingMethod>
      <ShippingMethodId>US Standard</ShippingMethodId>
      <ServiceLevel>Standard</ServiceLevel>
      <Rate>
        <WeightBased>
          <Amount>3.49</Amount>
          <CurrencyCode>GBP</CurrencyCode>
        </WeightBased>
      </Rate>
      <IncludedRegions>
        <USZipRegion>98104</USZipRegion>
      </IncludedRegions>
    </ShippingMethod>
    <ShippingMethod>
      <ShippingMethodId>US Expedited</ShippingMethodId>
      <ServiceLevel>Expedited</ServiceLevel>
    </ShippingMethod>
  </ShippingMethods>
</OrderCalculationsResponse>
```
4.2.2 Excluding a Shipping Speed

To indicate that you don’t support a shipping speed for a particular buyer – for example, you don’t support one- or two-day shipping to Alaska (ZIP Code 99950) – simply don’t include the specific shipping speeds as the ShippingMethod in your response.

For example, giving the response code above, your ShippingMethods section might include just US Expedited but not OneDay or TwoDay shipping, and it would look like this:

```
<ShippingMethods>
  <ShippingMethod>
    <ShippingMethodId>US Standard</ShippingMethodId>
    <ServiceLevel>Standard</ServiceLevel>
    <Rate>
      <WeightBased>
        <Amount>3.49</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </WeightBased>
      <IncludedRegions>
        <USZipRegion>98104</USZipRegion>
      </IncludedRegions>
    </Rate>
  </ShippingMethod>
</ShippingMethods>
```
4.2.3 Excluding an Address Entirely
To indicate that you don’t support shipping to a buyer – for example, if the buyer lives in Honolulu, Hawaii (ZIP Code 96720) – simply add the ExcludedRegions tag set with the same value as the IncludedRegions tag as shown below in this example.

Note
Even though you won’t ship to a region, you must still send us the Rate information, as this information is always required by the schema.

4.3 Adding an Error Response
If you cannot compute a required element and want to indicate an error, you can add the Error element to your response instead of including any CallbackOrders information. We then evaluate what to do with your response depending upon the value of the

If you send an error response INTERNAL_SERVER_ERROR or SERVICE_UNAVAILABLE, then we will process the order based on the value of the ProcessOrderOnCallbackFailure tag. If the tag is set to “true”, we will process the order using the overrides in the cart XML and the settings in Seller Central. If it’s set to “false”, then we will halt processing the order and display an error message to the buyer.

If you send the INVALID_SHIPPING_ADDRESS error response, we will prompt the buyer to pick another shipping address.

4.3.1 Error Response Example
An example of a valid error response appears below:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OrderCalculationsResponse xmlns="Checkout by Amazon Shopping Cart">
  <Response>
    <Error>
      <Code>INTERNAL_SERVER_ERROR</Code>
      <Message>Under Maintenance</Message>
    </Error>
  </Response>
</OrderCalculationsResponse>
```

4.3.2 Error Response Definitions
Below are the error codes returned to your webservice when there is an error in your response.

<table>
<thead>
<tr>
<th>Code</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use</td>
<td>XML</td>
</tr>
<tr>
<td>Definition</td>
<td>The error code for your response.</td>
</tr>
<tr>
<td>Required</td>
<td>Yes, if Error is sent as the response.</td>
</tr>
<tr>
<td>XPath</td>
<td>/MerchantCalculationResponse/Response/Error/Code</td>
</tr>
<tr>
<td>Type</td>
<td>Enumerated String</td>
</tr>
<tr>
<td>Value/Range</td>
<td>INVALID_SHIPPING_ADDRESS</td>
</tr>
</tbody>
</table>
| Example                  | <Response>
  <Error>
    <Code>INTERNAL_SERVER_ERROR</Code>
    <Message>Under Maintenance</Message>
  </Error>
</Response>
### Error

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The container element for the error information in your response.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Either CallbackOrders or Error is required in a response.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/MerchantCalculationResponse/Response/Error</td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
<td>Code, Message</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
</tbody>
</table>

#### Example

```xml
<Response>
  <Error>
    <Code>INTERNAL_SERVER_ERROR</Code>
    <Message>Under Maintenance</Message>
  </Error>
</Response>
```

### Message

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>A fuller explanation of the error. The buyer does not see this message.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Either CallbackOrders or Error is required in a response.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/MerchantCalculationResponse/Response/Error/Message</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>40-character string</td>
</tr>
</tbody>
</table>

#### Example

```xml
<Response>
  <Error>
    <Code>INTERNAL_SERVER_ERROR</Code>
    <Message>Under Maintenance</Message>
  </Error>
</Response>
```

### 4.4 How the ProcessOrderOnCallbackFailure Tag Is Used

The `ProcessOrderOnCallbackFailure` entry in the `MerchantCalculationCallback` section of the cart XML determines how we process the order when the response from your webservice is invalid or missing. If you do not specify the `ProcessOrderOnCallbackFailure` entry in your cart XML, then the value is implicitly set to “true”; you must explicitly set it to “false” in your cart XML if you want the order processing to stop when your response to the callback request is invalid or missing.
The ProcessOrderOnCallbackFailure flag is evaluated when callback fails for any of the following reasons:

<table>
<thead>
<tr>
<th>Error</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cannot reach merchant endpoint.</td>
<td>You specified a bad host name for your endpoint or your webservice is down.</td>
</tr>
<tr>
<td>Timeout reaching merchant endpoint.</td>
<td>Your webservice did not respond within five seconds.</td>
</tr>
<tr>
<td>Missing endpoint.</td>
<td>Your cart XML is missing the endpoint URL.</td>
</tr>
<tr>
<td>Merchant explicitly specified “Error” as a response.</td>
<td>You sent “Error” as your response and you specified either INTERNAL_SERVER_ERROR or SERVICE_UNAVAILABLE.</td>
</tr>
<tr>
<td>Response from the merchant is invalid because of missing or extra information.</td>
<td>Your response has too few or too many elements.</td>
</tr>
<tr>
<td>Merchant response is invalid because of a bad schema.</td>
<td>Your response isn’t using the right XMLNS URL or it specifies the cart XML incorrectly.</td>
</tr>
<tr>
<td>Merchant response is not in a valid format.</td>
<td>Your response is not using key-value pairs, is not URL-encoded, has a bad signature, or is missing a required element.</td>
</tr>
</tbody>
</table>

If you set ProcessOrderOnCallbackFailure to “true”, then order-processing continues even if your response is missing or invalid. Promotional discounts and shipping rates are calculated using either the information in your cart XML (the overrides) or the settings in Seller Central.
### 4.5 Order Processing on Fallback Failure Table

Here’s a table that shows how an order is processed depending on various states of the callback request and your webservice’s response.

<table>
<thead>
<tr>
<th>Calculate Shipping Rates</th>
<th>Calculate Promotions</th>
<th>Shipping Methods</th>
<th>Promotions</th>
<th>Fallback Failure</th>
<th>Response Status</th>
<th>Order Processed</th>
<th>Reason for Success or Failure</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>Valid</td>
<td>Yes</td>
<td>Everything requested is supplied correctly in the response.</td>
</tr>
<tr>
<td>TRUE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>Valid</td>
<td>Yes</td>
<td>Everything requested is supplied correctly in the response.</td>
</tr>
<tr>
<td>FALSE</td>
<td>FALSE</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>Valid</td>
<td>Yes</td>
<td>No callback is made, because all two tags are set to “false”.</td>
</tr>
<tr>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>Invalid</td>
<td>Yes</td>
<td>Promotions provided in response by mistake, but order will still be processed with overrides in cart XML or with Seller Central settings, because fallback is “true”.</td>
</tr>
<tr>
<td>TRUE</td>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>Invalid</td>
<td>Yes</td>
<td>Shipping information was missing in response, but order will still be processed with overrides in cart XML or with Seller Central settings, because fallback is “true”.</td>
</tr>
<tr>
<td>TRUE</td>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
<td>Invalid</td>
<td>No</td>
<td>Order processing stops because response is invalid (missing Shipping) and fallback is “False”.</td>
</tr>
</tbody>
</table>
4.6 Error Messages the Buyer Sees
If your callback response fails, whether due to a problem with the message, the transmission, or a stated Error response, the ProcessOrderOnCallbackFailure flag is used to determine what to do next. If this flag is set to "false", then the order is not processed and the buyer sees an error message like the one displayed below:

![Important Message]

4.7 Error Responses You Receive
There is one key-value pair in the error response, with five entries for the value. Because the value is XML that “wraps” other XML code, the “wrapped” XML code is marked as XML CDATA.

<table>
<thead>
<tr>
<th>Entry</th>
<th>Definition</th>
<th>Inclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. OrderCalculationsError</td>
<td>The container for the complete error response</td>
<td>Always included</td>
</tr>
<tr>
<td>2. OrderCalculationsErrorCode</td>
<td>The literal code describing the error</td>
<td>Always included</td>
</tr>
<tr>
<td>3. OrderCalculationsErrorMessage</td>
<td>The literal message explaining the error</td>
<td>Always included</td>
</tr>
<tr>
<td>4. OrderCalculationsRequest</td>
<td>What you received from our webservice</td>
<td>Always included</td>
</tr>
<tr>
<td>5. OrderCalculationsResponse</td>
<td>What you sent in response</td>
<td>Always included</td>
</tr>
</tbody>
</table>

Note that order-calculations-error is all lowercase letters. Note that the order of the response is as follows:

```
order-calculations-error=[OrderCalculationsError OrderCalculationsErrorCode OrderCalculationsErrorMessage OrderCalculationsRequest OrderCalculationsResponse]
```

Also note that the error response is not URL-encoded.

You can parse this response to retrieve the error code and error message, then use the error code/message table below to identify the issue. We return the request & response to help you solving the problem.
### 4.7.1 Error Codes and Messages

This table lists the error codes and the messages you might receive.

<table>
<thead>
<tr>
<th>Error</th>
<th>ErrorMessage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AdditionalInformationError</strong></td>
<td>The OrderCalculationsResponse contained MORE data than was specified in the OrderCalculationsRequest. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td><strong>AdditionalInformationError-Fallback</strong></td>
<td>The OrderCalculationsResponse contained MORE data than was specified in the OrderCalculationsRequest. However, default calculations were applied per the ProcessOrderOnCallbackFailure tag. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td><strong>HttpException</strong></td>
<td>An exception occurred while contacting the Order Calculations webservice. This typically constitutes an intermittent exception. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td><strong>HttpException-Fallback</strong></td>
<td>An exception occurred while contacting the Order Calculations webservice. This typically constitutes an intermittent exception. However, default calculations were applied per the ProcessOrderOnCallbackFailure tag. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td><strong>HttpTimeoutError</strong></td>
<td>A timeout occurred while trying to reach the Order Calculations webservice. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td><strong>HttpTimeoutError-Fallback</strong></td>
<td>A timeout occurred while trying to reach the Order Calculations webservice. However, default calculations were applied per the ProcessOrderOnCallbackFailure tag. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td><strong>InvalidEndpointError</strong></td>
<td>The merchant specified was invalid. Supported endpoints are either https or http for Sandbox and only https for Production. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td><strong>InvalidEndpointError-Fallback</strong></td>
<td>The merchant endpoint specified was invalid. Supported endpoints are either http or https for Sandbox and only https for Production. However, default calculations were applied per the ProcessOrderOnCallbackFailure tag. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td><strong>InvalidSSLCertificateError</strong></td>
<td>The URL endpoint specified appears to have an invalid or expired SSL certificate. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>Error</td>
<td>ErrorMessage</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>InvalidSSLCertificateError</td>
<td>The URL endpoint specified appears to have an invalid or expired SSL certificate. However, default calculations were applied per the ProcessOrderOnCallbackFailure tag. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>MerchantSpecifiedError</td>
<td>The Order Calculations webservice returned an explicit error. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>MerchantSpecifiedError-Fallback</td>
<td>The Order Calculations webservice returned an explicit error. However, default calculations were applied per the ProcessOrderOnCallbackFailure tag. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>MissingInformationError</td>
<td>The OrderCalculationsResponse contained LESS data than was specified in the OrderCalculationsRequest. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>MissingInformationError-Fallback</td>
<td>The OrderCalculationsResponse contained LESS data than was specified in the OrderCalculationsRequest. However, default calculations were applied per the ProcessOrderOnCallbackFailure tag. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>ResponseOverflowError</td>
<td>The OrderCalculationsResponse exceeded the maximum possible size. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>ResponseOverflowError-Fallback</td>
<td>The OrderCalculationsResponse exceeded the maximum possible size. However, default calculations were applied per the ProcessOrderOnCallbackFailure tag. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>ResponseValidationError</td>
<td>The OrderCalculationsResponse could not be validated. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>ResponseValidationError-Fallback</td>
<td>The OrderCalculationsResponse could not be validated. However, default calculations were applied per the ProcessOrderOnCallbackFailure tag. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>SignatureMismatchError</td>
<td>The OrderCalculations webservice returned an invalid signature. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>Error</td>
<td>ErrorMessage</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SignatureMismatchError-Fallback</td>
<td>The Order Calculations webservice returned an invalid signature. However, default calculations were applied per the ProcessOrderOnCallbackFailure tag. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
<tr>
<td>PromoBusinessRulesViolationError</td>
<td>The promotional discount rate must be greater than or equal to 0.0 and less than or equal to 1.0. To debug this error, please refer to the OrderCalculationsRequest or OrderCalculationsResponse for more details.</td>
</tr>
</tbody>
</table>
5 Using Custom Data Fields with Callbacks

You can use your own data definitions and data fields in your Order XML feeds. For example, if you want to allow customers to personalize the mugs and cups you sell, you can pass the custom text field “cust_name” in your Order XML. When we send the callback request to you, we will include this information.

Note that when you send your callback response, you do not need to re-send this custom data.

5.1 Prerequisites for Using Custom Data Fields

To use custom data fields in your feeds, your XML-based applications must meet the following conditions:

2. You must be using the order.xsd version with a file date of 2009-03-04 or later. (Download this version here.)
3. You must be using the callback.xsd version with a file date of 2009-03-04 or later. (Download this version here.)

5.2 How We Process the Custom Data Fields

When we receive your order XML, your order notification response, or your callback response which contains your custom data, we simply store it until we return it to you as part of the Callback response.

We do not parse the data or attempt to interpret it. As long as the content is syntactically correct (tags spelled the same for both opening and closing tags, no tags improperly nested, and so on), we accept the content as-is, and return it to you in the same way.

Because we do not parse the custom data you send that's marked with the ItemCustomData or CartCustomData tags, you do not need to send us an XSD or otherwise define your custom data fields.

5.3 Modifying Your Order XML to Use Custom Data Fields

To use custom data fields in your order, you add a new section in your order XML tagged with ItemCustomData or CartCustomData, as shown in the following example:

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <Cart>
    <Items>
      <Item>
        <SKU>JKL909</SKU>
        <MerchantId>AEIOU1234AEIOU</MerchantId>
        <Title>Calvin and Hobbes Reliquary</Title>
        <Price>
          <Amount>29.99</Amount>
        </Price>
      </Item>
    </Items>
  </Cart>
</Order>
```
We’ll return it to you as part of the Callback API.

5.4 Using Custom Data Fields with the Callback API

To use custom data fields with callbacks, you enter the information (as shown above) in the ItemCustomData or CartCustomData areas.

When we send you the Callback Request XML, we include this information in the file, as shown in the example below:

```xml
<?xml version="1.0" encoding="UTF-8"?>
  CallbackReferenceId="1-8f6fc8e5-c583-47fe-af44-3a05abd875a2"
  OrderCalculationCallbacks="true">
  <OrderCallbackEndpoint>localhost</OrderCallbackEndpoint>
</OrderCalculationsRequest>
<Item
  SKU="JKL909"
  MerchantId="AEIOU1234AEIOU"
  Title="Calvin and Hobbes Reliquary"
  Price="29.99"
  Amount="GBP"
  CurrencyCode="GBP"
  ItemCustomData="true">
  <CustomText>
    <Text>Charles River Sports Academy</Text>
    <Color>Malachite</Color>
    <StyleID>10000</StyleID>
  </CustomText>
</Item>
```
When you receive this data, you can manipulate it using your own application logic to process the order and return the required information as part of your callback response.

5.5 Using a Custom Namespace as Part of the Custom Data Entry

You can provide your own namespace as part of your custom data entry.

Note the added XMLNS in the example below, listed as CustomText
xmlns:mynamespace="http://mydomain.com":

```
<?xml version="1.0" encoding="UTF-8"?>
  <Cart>
    <Items>
      <Item>
        <SKU>JKL909</SKU>
        <MerchantId>AEIOU1234AEIOU</MerchantId>
        <Title>Calvin and Hobbes Reliquary</Title>
        <Price>
          <Amount>29.99</Amount>
          <CurrencyCode>GBP</CurrencyCode>
        </Price>
        <Quantity>1</Quantity>
        <Weight>
          <Amount>8.5</Amount>
          <Unit>kg</Unit>
        </Weight>
        <Category>Books</Category>
        <ItemCustomData>
          <mynamespace:CustomText xmlns:mynamespace="http://mydomain.com">
            <mynamespace:Text>Charles River Sports Academy</mynamespace:Text>
            <mynamespace:Color>Malachite</mynamespace:Color>
            <mynamespace:StyleID>10090</mynamespace:StyleID>
          </mynamespace:CustomText>
        </ItemCustomData>
      </Item>
    </Items>
    <CartCustomData>
      <CartNumber>0525470948</CartNumber>
      <CartCustomData>
    </Cart>
    <CallbackOrders>...
  </CallbackOrders>
</Order>
```

Because you added this custom XMLNS, the responses are modified, too.
Here is an example of the resulting callback:

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <CallbackReferenceId>1-8f6fc8e5-c583-47fe-a944-3a05abd875a2</CallbackReferenceId>
  <OrderCalculationCallbacks>
    <CalculateShippingRates>true</CalculateShippingRates>
    <OrderCallbackEndpoint>localhost</OrderCallbackEndpoint>
  </OrderCalculationCallbacks>
  <Cart>
    <Items>
      <Item>
        <SKU>JKL909</SKU>
        <MerchantId>AEIOU1234AEIOU</MerchantId>
        <Title>Calvin and Hobbes Reliquary</Title>
        <Price>
          <Amount>29.99</Amount>
          <CurrencyCode>GBP</CurrencyCode>
        </Price>
        <ItemCustomData>
          <mynamespace:CustomText xmlns:mynamespace="http://mydomain.com">
            <mynamespace:Text>Charles River Sports Academy</mynamespace:Text>
            <mynamespace:Color>Malachite</mynamespace:Color>
            <mynamespace:StyleID>10090</mynamespace:StyleID>
          </mynamespace:CustomText>
        </ItemCustomData>
      </Item>
    </Items>
    <CartCustomData>
      <CartNumber>0525470948</CartNumber>
      Customer <CName>John Smith</CName> wants delivery only in the evenings
    </CartCustomData>
  </Cart>
  <CallbackOrders>
    ...
  </CallbackOrders>
</OrderCalculationsRequest>
```

Note that the callback response you send will not have any ItemCustomData or CartCustomData.
6 Using Real-time Cart Updates with the Callback API

Real-time Cart Updates let you dynamically add or remove products and surcharges in the shopping cart based upon factors such as inventory availability or the customer’s shipping address. For example, suppose a customer selects a high-velocity product and an electronic display device such as a television set. Then, while going through the checkout pipeline, the high-velocity product becomes out-of-stock, and the customer selects a California shipping address. With Real-time Cart Updates, you can dynamically remove the out-of-stock product from the cart and add a fee surcharge for the television set to comply with the California Electronic Waste Recycling Fee (sometimes called the “Green Fee”). This allows the customer to complete the purchase (only the out-of-stock product is removed) and for the merchant to collect the required fees that are based on the shipping address.

6.1 How Real-time Cart Updates Works

1. The buyer begins the checkout process.
2. You send the XML containing the purchase information (and with the Callback API enabled) to Amazon Payments.
3. While in the checkout pipeline, the buyer selects a shipping address in real time.
4. We make a Merchant Callback Request to your Merchant Callback Webservice. The request contains the XML from step #2 and the shipping address from step #3.
5. Your callback webservice must then determine the appropriate calculations for each item in the XML request we send. If the shipping address or item availability matches specific criteria (such as a California shipping address or an item out-of-stock), then you ADD or REMOVE a fee or item from the cart.

Notes

The buyer is not explicitly notified about the changes to the shopping cart, although she can see the changes to the shopping cart each time she changes the shipping address or a product is automatically removed.

The buyer can keep repeating step #3 (select a different shipping address); each time she does so, we send a callback request. Your webservice should then re-evaluate the purchase and submit an updated callback response.

A bad callback response for Real-time Cart Updates is handled as any other bad callback response. See How the ProcessOrderOnCallbackFailure Tag Is Used for more information.

6.2 Enabling Real-time Cart Updates in Your Cart XML

You have two tasks to accomplish to enable Real-time Cart Updates to your cart XML:

1. Add the code snippets to set the surcharge or remove the item
2. Override the shipping method for the surcharge (where appropriate)
Notes

If the product you remove would enable a promotion, then the promotion will no longer apply. That is, if you require a hat for a promotional discount on gloves, but you remove the hat with the REMOVE value, then the promotional discount will no longer apply.

Promotions can apply to any item in the cart, including a surcharge. If you specify promotions, be sure you set the promotion amount for surcharges appropriately—in most cases, the promotion amount should be set to zero.

6.2.1 Add the Code to Enable Real-time Cart Updates

To enable Real-time Cart Updates, you add the following code to your response XML:

```xml
...<UpdatedCartItems>
  <UpdatedCartItem>
    <SKU>GREEN-FEE</SKU>
    <MerchantId>AEIOU1234AEIOU</MerchantId>
    <Title>EWRF15-CA Imposed Electronic Waste Recycling Fee: 15-34.99-Inch</Title>
    <Price>
      <Amount>15.00</Amount>
      <CurrencyCode>GBP</CurrencyCode>
      <Quantity>1</Quantity>
    </Price>
    <UpdateType>ADD</UpdateType>
  </UpdatedCartItem>
</UpdatedCartItems>
...```

Note

Your XML-based cart must already be enabled to use the Callback API; see [Enable Callbacks in Your Cart XML](#), above, for more information.

The UpdateType field has two acceptable values: ADD (to add a surcharge to the cart) and REMOVE (to remove an item from the cart).

When your webservice sends the callback response with this information, the shopping cart is updated and the customer can review and approve the order before placing the order.

6.2.2 Override the Shipping Rates for Surcharges

In most cases, you must override the shipping rates for a “Green Fee” surcharge, because this fee is not a physical product (and has no shipping requirement).

When you override the surcharge shipping rates, you should be careful to observe the following:

1. To keep the “Green Fee” surcharge and the surcharge-eligible item together on the same order, the surcharged item and the surcharge fee must use the same shipping method. Otherwise, it's possible for a split order to cause the surcharge to appear on one shipment, and the surcharge-eligible item to appear on another shipment.
2. To keep the “Green Fee” surcharge from being included in the shipping rate calculations, the shipping methods used for the purchase must either be shipment-based (for the entire order), or weight-based (with no Weight attribute for the surcharge). The order should not use item-quantity—based shipping.

For example, suppose you sell a 19-inch LCD TV with a “Green Fee” surcharge of $16 for shipments to California. The LCD incurs the standard shipping rate (standard-shipping-1).

```xml
<?xml version="1.0"?>
  <Response>
    <CallbackOrders>
      <CallbackOrder>
        <Address>
          <AddressId>b26a2067266403ad46e73470c86508b0</AddressId>
        </Address>
        <CallbackOrderItems>
          <CallbackOrderItem>
            <SKU>LCD-MONITOR-19</SKU>
            <ShippingMethodIds>
              <ShippingMethodId>standard-shipping-1</ShippingMethodId>
            </ShippingMethodIds>
          </CallbackOrderItem>
        </CallbackOrderItems>
        <UpdatedCartItems>
          <UpdatedCartItem>
            <SKU>GREEN-FEE</SKU>
            <MerchantId>AEIOU1234AEIOU</MerchantId>
            <Title>EWRF15-CA Imposed Electronic Waste Recycling Fee: 15-34.99-Inch</Title>
            <Price>
              <Amount>15.00</Amount>
              <CurrencyCode>GBP</CurrencyCode>
            </Price>
            <Quantity>1</Quantity>
            <ShippingMethodIds>
              <ShippingMethodId>standard-shipping-1</ShippingMethodId>
            </ShippingMethodIds>
            <UpdateType>ADD</UpdateType>
          </UpdatedCartItem>
        </UpdatedCartItems>
      </CallbackOrder>
    </CallbackOrders>
  </Response>
</OrderCalculationsResponse>
```
</OrderCalculationsResponse>
7 Using Callbacks in the Sandbox

In the Production environment, Checkout by Amazon makes callbacks only over HTTPS connections, using port 443. In the Sandbox environment, Checkout by Amazon makes calls over HTTP connections using port 80.

Callbacks behave the same way in Sandbox as in Production. The responses are validated and merged into the original order if the response is valid. None of the orders in the Sandbox environment are processed.
8  Frequently Asked Questions About the Callback API

8.1  What Is the Callback API?
The Callback API enables you to calculate your own shipping costs or promotional discounts using your own services at the time the customer places the order. The shipping address selected by the buyer is sent to your Order Calculation Callback URL; we will then use your response to this request to charge the customer the shipping charges and the promotional discount your service calculated.

8.2  What Do I Need to Do to Integrate with the Callback API?
The Callback API is designed for registered Checkout by Amazon merchants. If you are not already signed up for this product, please go to our Sign Up page at https://payments.amazon.com/sdui/sdui/business?sn=cba/o.

After you sign up, please review the Integration Guide and Developer Guide to learn how to integrate your website with Checkout by Amazon as well as with the Callback API.

If you are an existing Checkout by Amazon merchant, then please review the Implementation Guides, found in the Technical Resources help section, to integrate your website with the Callback API. You'll use the same order.xsd to create your cart XML and your response XML, so you'll find the XML definitions useful.

8.3  How Do I Set Up Shipping Charges?
You can set up shipping charges using one of these methods:

1. Go to Seller Central, and click the Settings tab, then click the Shipping Settings link to view the current shipping charges. You can also make changes by clicking Edit the Current Configuration.
2. Pass us shipping rules (overrides) in the order you send to Amazon Payments. See the Implementation Guides, found in the Technical Resources help section, for more information.
3. Calculate and send us the shipping charges in response to our callback request.

Note
Shipping charges are exclusive. If you set up shipping rates in Seller Central, and send shipping overrides in the order you send to Amazon Payments, and then send us shipping charges in response to our callback request, we evaluate the shipping charges in this order: first, we use the shipping charges in your callback request. If we can't use the callback request, and you specify ProcessOrderOnCallbackFailure as “true”, then we use the shipping overrides in your order; then, if you also did not send a shipping override, we use your shipping rates set in Seller Central.
8.4 How Do I Manage Promotions?

You can set up promotions using one of these methods:

1. Go to Seller Central, and click the Settings tab. Click the Manage Promotions link to view the current promotions. From this page, you can Add New Promotions or Manage Product Lists.
2. Pass us promotions rules (overrides) in the order you send to Amazon Payments. See the Implementation Guides, found in the Technical Resources help section, for more information.
3. Calculate and send us the promotional discounts in response to our callback request.

**Note**

Promotional benefits are additive. If you set promotions in Seller Central and send a promotion override, and then send a promotional discount in response to the callback request, we will evaluate all applicable promotional benefits when calculating the total benefit amount in this order: first, we use the promotion discounts in your callback request. If we can’t use the callback request and you specify ProcessOrderOnCallbackFailure as “true”, then we use the promotion overrides in your order and the promotions you set in Seller Central.

8.5 Are Item SKUs Mandatory When Using the Callback API?

Yes, Item SKUs are mandatory when using the Callback API.

For example, each Item in your response should have a SKU entry that looks similar to this:

```xml
...<Item><SKU>JKL909</SKU><MerchantId>AEIOU1234AEIOU</MerchantId><Title>Calvin and Hobbes Reliquary</Title><Price><Amount>29.99</Amount><CurrencyCode>GBP</CurrencyCode></Price><Quantity>1</Quantity></Item>...
```

When you send your callback response, you'll enter the individual SKU in the ShipItem section instead of the Item section.

8.6 What Should I Expect If the Callback Fails?

If the callback fails (for example, your endpoint URL is not reachable, you fail to respond within five seconds of the request, or your request is malformed), then the order is processed depending upon the ProcessOrderOnCallbackFailure flag in the cart XML. If this flag is set to “true” or this element is missing from the cart XML, then the order is processed using any shipping or promotion override information in the cart; if these overrides are missing from the cart XML, then the order is processed using the settings in Seller Central.
If the `ProcessOrderOnCallbackFailure` flag is set to “false”, then the processing stops and the order is rejected. Your buyer sees an error message.

If you build carts that use the Callback API, you should consider that errors might occur from time to time. For example, your application might not respond in time, or your response might be malformed. If you have set the `ProcessOrderOnCallbackFailure` flag in the cart XML to “true”, then we’ll look at your override information (if any) in your cart XML, and then we’ll use your settings (if any) in Seller Central. If you don’t send overrides in your cart XML and you haven’t modified your settings in Seller Central, your order charges might not be what you expect. For example, by default Seller Central has no promotions set up; you must create them before they can be used to calculate discounts for your orders. (Note that promotions are additive; if your callback fails but you specified `ProcessOrderOnCallbackFailure` as “true”, then we’ll use both the promotion overrides and the promotions in Seller Central, if any, to calculate the promotional benefit.)

If you send malformed content (such as shipping rates when we don’t request them, or not sending them when we do request them), none of the `OrderCalculationResponse` values are processed – it’s an all-or-nothing situation.

### 8.7 Why Am I Not Receiving a Callback Request?

A callback request is generated for your cart XML only if it meets the following conditions:

2. You specified the callback endpoint using https:// in Production. (You can use either http:// or https:// when using the Sandbox.)
3. Your cart XML includes a SKU for every specified Item.
4. You have an `OrderCalculationCallbacks` section in your cart XML similar to the following (your `OrderCallbackEndpoint` will differ).

   ```xml
   <OrderCalculationCallbacks>
   <CalculatePromotions>true</CalculatePromotions>
   <CalculateShippingRates>true</CalculateShippingRates>
   <OrderCallbackEndpoint>https://my.endpoint.com/receive.php</OrderCallbackEndpoint>
   <ProcessOrderOnCallbackFailure>true</ProcessOrderOnCallbackFailure>
   </OrderCalculationCallbacks>
   ```

### 8.8 When Should I Use Callbacks?

You can use callbacks when you want to use your own applications to calculate shipping or promotional discounts. You can use this if you want to simplify the creation of your XML-based carts; you can create a much smaller code snippet for your cart and perform the calculations when the order is placed, saving you much time in development and debugging.

### 8.9 Can I Use the Callback API If I Use HTML-based Carts?

No. The Callback API is only for carts you build yourself using XML.
8.10 Can I Use the Callback API If I Use the Button Generator in Seller Central?
No. The Callback API is only for carts you build yourself using XML.

8.11 What Happens If My Cart Response Does Not Match the Callback Request?
If the callback fails (for example, your endpoint URL is not reachable, you fail to respond within five seconds of the request, or your request is malformed), then the order is processed depending upon the ProcessOrderOnCallbackFailure flag in the cart XML. If this flag is set to “true” or this element is missing from the cart XML, then the order is processed using any shipping or promotion override information in the cart; if these overrides are missing from the cart XML, then the order is processed using the settings in Seller Central.

If the ProcessOrderOnCallbackFailure flag is set to “false”, then the processing stops and the order is rejected. Your buyer sees an error message.

Note that if you send malformed content (such as shipping rates when we don't request them, or not sending them when we do request them), none of the OrderCalculationResponse values are processed – it’s an all-or-nothing situation.

8.12 Must the ShippingMethodIds or PromotionIds Match the Cart Values?
No. Merchant can specify different IDs in the response and Checkout by Amazon will use the response IDs.

8.13 What Should I Look For When My Callback Response Is Failing?
If you send malformed content (such as shipping rates when we don't request them, or not sending them when we do request them), none of the OrderCalculationResponse values are processed – it’s an all-or-nothing situation. Instead, we will use the values in your overrides (if any) or your settings in Seller Central (if any).

You can look at your logs to find error notification messages to get more information.

8.14 Must I Include the IncludedRegion Entry When I Specify an Excluded Address?
Yes. You must include the IncludedRegion in your response, as that entry is required by the XSD. It’s easiest just to set it to the same value as the ExcludedRegion. So if you are excluding the ZIP Code 98104, your code would appear as follows:

```
<IncludedRegions><USZipRegion>98104</USZipRegion></IncludedRegions>
<ExcludedRegions><USZipRegion>98104</USZipRegion></ExcludedRegions>
```

8.15 What Is the Maximum Length of the URL?
The URLs your specify in your response cannot be longer than 1000 characters. A URL longer than 1000 characters is truncated at the 1000th character. This limitation applies to the following URLs:

- OrderCallbackEndpoint
- ReturnUrl
• CancelUrl
• YourAccountUrl
## Appendix A: The Callback Schema Definitions

Here is the complete list of Callback code elements and their definitions for orders processed by Amazon Payments.

### Note

Case is significant for XML.

---

### A

<table>
<thead>
<tr>
<th><strong>Address</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>The container for the address information.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
</tbody>
</table>
| **XPath**   | /OrderCalculationsRequest/CallbackOrders/CallbackOrder/Address  
/OrderCalculationsResponse/Response/CallbackOrders/CallbackOrder/Address |
| **Required subentries** | AddressId, City, State, PostalCode, CountryCode |
| **Optional subentries** | AddressOne, AddressTwo, AddressThree |
| **Type**    | Container |
| **Example** | <Address>  
  <AddressId>AddressId</AddressId>  
  <AddressFieldOne>1200 5th Avenue South</AddressFieldOne>  
  <AddressFieldTwo>Suite 116</AddressFieldTwo>  
  <AddressFieldThree>Station M162K7</AddressFieldThree>  
  <City>Seattle</City>  
  <State>WA</State>  
  <PostalCode>98104</PostalCode>  
</Address> |
### AddressFieldOne, AddressFieldTwo, AddressFieldThree

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The description (or abbreviation) for the CallbackOrder method.</td>
</tr>
</tbody>
</table>
| **Required** | AddressFieldOne: Yes  
                        AddressFieldTwo, AddressFieldThree: No |
| **XPath** | /OrderCalculationsRequest/CallbackOrders/CallbackOrder/Address/AddressFieldOne  
                          /OrderCalculationsRequest/CallbackOrders/CallbackOrder/Address/AddressFieldTwo  
                          /OrderCalculationsRequest/CallbackOrders/CallbackOrder/Address/AddressFieldThree |
| **Type** | String |
| **Value/Range** | 40-character string |
| **Example** | `<Address>`  
                        <AddressId>AddressId</AddressId>  
                        <AddressFieldOne>1200 5th Avenue South</AddressFieldOne>  
                        <AddressFieldTwo>Suite 116</AddressFieldTwo>  
                        <AddressFieldThree>Station M162K7</AddressFieldThree>  
                        <City>Seattle</City>  
                        <State>WA</State>  
                        <PostalCode>98104</PostalCode>  
                        </Address> |

### AddressId

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>Unique identifier for the buyer's address.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
</tbody>
</table>
| **XPath** | /OrderCalculationsRequest/CallbackOrders/CallbackOrder/Address/AddressId  
                          /OrderCalculationsResponse/Response/CallbackOrders/CallbackOrder/Address/AddressId |
| **Type** | String |
| **Value/Range** | 40-character string |
| **Example** | `<Address>`  
                        <AddressId>AddressId</AddressId>  
                        <AddressFieldOne>1200 5th Avenue South</AddressFieldOne>  
                        <AddressFieldTwo>Suite 116</AddressFieldTwo>  
                        <AddressFieldThree>Station M162K7</AddressFieldThree>  
                        <City>Seattle</City>  
                        <State>WA</State>  
                        <PostalCode>98104</PostalCode>  
                        </Address> |
<table>
<thead>
<tr>
<th><strong>Amount</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
</tr>
<tr>
<td><strong>XML</strong></td>
</tr>
</tbody>
</table>
| **Definition** | The discount amount expressed as a percentage of the Price (DiscountRate). A discount rate of 6.5% is expressed as 0.0650, not 6.5.  
- OR -  
The discount amount expressed as units of currency (FixedAmountDiscount). |
| **Required** | Yes, if CalculatePromotions is “true” in the cart XML and the discount type is fixed rate. |
| **XPath** | /OrderCalculationsResponse/Promotions/Promotion/Benefit/DiscountRate/Amount  
- OR -  
/OderCalculationsResponse/Promotions/Promotion/Benefit/FixedAmountDiscount/Amount |
| **Type** | Positive double |
| **Value/Range** | 0.0001-0.9999 (DiscountRate)  
0.01-99999.99 (FixedAmountDiscount) |
| **Example** | `<Promotions>`  
  `<Promotion>`  
   `<PromotionId>TenPound</Description>`  
   `<Description>Ten Pounds Off</Description>`  
   `<Benefit>`  
    `<FixedAmountDiscount>`  
     `<Amount>10.00</Amount>`  
     `<CurrencyCode>GBP</CurrencyCode>`  
    `</FixedAmountDiscount>`  
   `</Benefit>`  
  `</Promotion>`  
`</Promotions>` |
### Amount

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The shipping charge amount expressed as units of currency. The shipping charge is calculated by multiplying the Amount by the Weight (WeightBased) or Quantity (ItemQuantityBased).</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if CalculatePromotions is “true” in the cart XML.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsResponse/ShippingMethods/ShippingMethod Rate/WeightBased/Amount</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Positive double</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>0.01-99999.99</td>
</tr>
</tbody>
</table>
| **Example** | `<ShippingMethods>
   <ShippingMethod>
     <ShippingMethodId>UPSGround</ShippingMethodId>
     <ServiceLevel>Standard</ServiceLevel>
     <Rate>
       <WeightBased>
         <Amount>3.49</Amount>
         <CurrencyCode>GBP</CurrencyCode>
       </WeightBased>
       <Rate>
         <IncludedRegions>
           <USZipRegion>98104</USZipRegion>
         </IncludedRegions>
       </Rate>
     </ShippingMethod>
   </ShippingMethods>` |

### Benefit

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The container for the promotion specified in the document.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if CalculatePromotions is “true” in the cart XML.</td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
<td>FixedAmountDiscount</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsResponse/Promotions/Promotion/Benefit</td>
</tr>
</tbody>
</table>
| **Example** | `<Promotions>
   <Promotion>
     <PromotionId>TenPound</Description>
     <Description>Ten Pounds Off</Description>
     <Benefit>
       <FixedAmountDiscount>
         <Amount>10.00</Amount>
         <CurrencyCode>GBP</CurrencyCode>
       </FixedAmountDiscount>
     </Benefit>
   </Promotion>
 </Promotions>` |
### CalculatePromotions

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>Stipulates whether you will calculate your own promotional discounts for the order. Set this to “true” if you will calculate the promotional discounts for the order; set to “false” if you want Amazon Payments to use the promotions overrides in your cart XML or the settings in Seller Central to calculate the promotional discounts. If you do not include this entry, the value is set to “false”. The CalculatePromotions entry must be specified before the CalculateShippingRates entry.</td>
</tr>
<tr>
<td>Required</td>
<td>No</td>
</tr>
<tr>
<td>XPath</td>
<td>/OrderCalculationCallbacks/CalculatePromotions</td>
</tr>
<tr>
<td>Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Value/Range</td>
<td>true</td>
</tr>
</tbody>
</table>

### CalculateShippingRates

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>Stipulates whether you will calculate your own shipping rates for the order. Set this to “true” if you will calculate shipping rates for the order; set to “false” if you want Amazon Payments to use shipping overrides in your cart XML or the settings in Seller Central to calculate shipping rates. If you do not include this entry, the value is set to “false”.</td>
</tr>
<tr>
<td>Required</td>
<td>No</td>
</tr>
<tr>
<td>XPath</td>
<td>/OrderCalculationCallbacks/CalculateShippingRates</td>
</tr>
<tr>
<td>Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Value/Range</td>
<td>true</td>
</tr>
</tbody>
</table>
### CallbackOrder

<table>
<thead>
<tr>
<th><strong>Use</strong></th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The container element for the CallbackOrder information in your response.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsResponse/Response/CallbackOrders/CallbackOrder</td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
<td>Address, CallbackOrderItems</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
</tbody>
</table>

**Example**

```xml
<CallbackOrders>
  <CallbackOrder>
    <Address>
      <AddressId>CustomerAddress1</AddressId>
    </Address>
    <CallbackOrderItems>
      <CallbackOrderItem>
        <SKU>JKL909</SKU>
        <PromotionIds>
          <PromotionId>TenPound</PromotionId>
        </PromotionIds>
        <ShippingMethodIds>
          <ShippingMethodId>US Standard</ShippingMethodId>
          <ShippingMethodId>US NextDay</ShippingMethodId>
        </ShippingMethodIds>
      </CallbackOrderItem>
    </CallbackOrderItems>
  </CallbackOrder>
</CallbackOrders>
```
## CallbackOrderItem

<table>
<thead>
<tr>
<th><strong>Use</strong></th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The container for the specific information about an item in your response.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsResponse/CallbackOrderItems/CallbackOrderItem</code></td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
<td>SKU</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
</tbody>
</table>

**Example**

```xml
<CallbackOrders>
  <CallbackOrder>
    <Address>
      <AddressId>CustomerAddress1</AddressId>
    </Address>
    <CallbackOrderItems>
      <CallbackOrderItem>
        <SKU>JKL909</SKU>
        <PromotionIds>
          <PromotionId>TenPound</PromotionId>
        </PromotionIds>
        <ShippingMethodIds>
          <ShippingMethodId>US Standard</ShippingMethodId>
          <ShippingMethodId>US NextDay</ShippingMethodId>
        </ShippingMethodIds>
      </CallbackOrderItem>
    </CallbackOrderItems>
  </CallbackOrder>
</CallbackOrders>
```
<table>
<thead>
<tr>
<th><strong>CallbackOrderItems</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>The container for the shipping item information in your response.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsResponse/CallbackOrderItems</td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
<td>CallbackOrderItem</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
</tbody>
</table>
| **Example**           | `<CallbackOrders>`  
|                       |   `<CallbackOrder>`  
|                       |     `<Address>`  
|                       |       `<AddressId>CustomerAddress1</AddressId>`  
|                       |     `/Address>`  
|                       |   `<CallbackOrderItems>`  
|                       |     `<CallbackOrderItem>`  
|                       |       `<SKU>JKL909</SKU>`  
|                       |       `<PromotionIds>`  
|                       |         `<PromotionId>TenPound</PromotionId>`  
|                       |       `/PromotionIds>`  
|                       |     `<ShippingMethodIds>`  
|                       |       `<ShippingMethodId>US Standard</ShippingMethodId>`  
|                       |       `<ShippingMethodId>US NextDay</ShippingMethodId>`  
|                       |     `/ShippingMethodIds>`  
|                       |   `/CallbackOrderItems>`  
|                       | `/CallbackOrder>`  
|                       | `/CallbackOrders>` |
## CallbackOrders

<table>
<thead>
<tr>
<th><strong>Use</strong></th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The container element for the CallbackOrders information in your response.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Either CallbackOrders or Error is required in a response.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsResponse/Response/CallbackOrders</td>
</tr>
<tr>
<td><strong>Required subentry</strong></td>
<td>CallbackOrder</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
</tbody>
</table>
| **Example** | `<CallbackOrders>
   <CallbackOrder>
     <Address>
       <AddressId>CustomerAddress1</AddressId>
     </Address>
     <CallbackOrderItems>
       <CallbackOrderItem>
         <SKU>JKL909</SKU>
         <PromotionIds>
           <PromotionId>TenPound</PromotionId>
         </PromotionIds>
         <ShippingMethodIds>
           <ShippingMethodId>US Standard</ShippingMethodId>
           <ShippingMethodId>US NextDay</ShippingMethodId>
         </ShippingMethodIds>
       </CallbackOrderItem>
     </CallbackOrderItems>
   </CallbackOrder>
</CallbackOrders>` |

## CallbackReferenceId

<table>
<thead>
<tr>
<th><strong>Use</strong></th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The unique identifier for the callback request.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsRequest</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>40-character string</td>
</tr>
</tbody>
</table>
| **Example** | `<?xml version="1.0" encoding="UTF-8"?>
  <CallbackReferenceId>
    1-f1b06763-b30c-47de-ab95-b4d8f91ef5
  </CallbackReferenceId>
  ...
</MerchantCalculationRequest>` |
### City

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>The name of the buyer's city where the item will be shipped to.</td>
</tr>
<tr>
<td>Required</td>
<td>Yes</td>
</tr>
<tr>
<td>XPath</td>
<td><code>/OrderCalculationsRequest/CallbackOrders/CallbackOrder/Address/AddressId/City</code></td>
</tr>
<tr>
<td>Type</td>
<td>String</td>
</tr>
<tr>
<td>Value/Range</td>
<td>40-character string</td>
</tr>
</tbody>
</table>

**Example**

```xml
<Address>
  <AddressId>AddressId</AddressId>
  <AddressFieldOne>1200 5th Avenue South</AddressFieldOne>
  <AddressFieldTwo>Suite 116</AddressFieldTwo>
  <AddressFieldThree>Station M162K7</AddressFieldThree>
  <City>Seattle</City>
  <State>WA</State>
  <PostalCode>98104</PostalCode>
</Address>
```

### Code

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>The error code for your response.</td>
</tr>
<tr>
<td>Required</td>
<td>Yes, if Error is sent as the response.</td>
</tr>
<tr>
<td>XPath</td>
<td><code>/MerchantCalculationResponse/Response/Error/Code</code></td>
</tr>
<tr>
<td>Type</td>
<td>Enumerated String</td>
</tr>
<tr>
<td>Value/Range</td>
<td>INVALID_SHIPPING_ADDRESS</td>
</tr>
</tbody>
</table>

**Example**

```xml
<Response>
  <Error>
    <Code>INTERNAL_SERVER_ERROR</Code>
    <Message>Under Maintenance</Message>
  </Error>
</Response>
```
### CountryCode

<table>
<thead>
<tr>
<th><strong>Use</strong></th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The country code for the buyer where the Item will be shipped to.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsRequest/CallbackOrders/CallbackOrder/Address/AddressId/CountryCode</code></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>2-character string</td>
</tr>
</tbody>
</table>

**Example**

```
<Address>
  <AddressId>AddressId</AddressId>
  <AddressFieldOne>1200 5th Avenue South</AddressFieldOne>
  <AddressFieldTwo>Suite 116</AddressFieldTwo>
  <AddressFieldThree>Station M162K7</AddressFieldThree>
  <City>Seattle</City>
  <State>WA</State>
  <PostalCode>98104</PostalCode>
</Address>
```

### CurrencyCode

<table>
<thead>
<tr>
<th><strong>Use</strong></th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The type of currency for the FixedAmountDiscount. Currently, only GBP is accepted.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if FixedAmountDiscount is specified.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsResponse/Promotions/Promotion/Benefit/FixedAmountDiscount/CurrencyCode</code></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>3-character string</td>
</tr>
</tbody>
</table>

**Example**

```
<Promotions>
  <Promotion>
    <PromotionId>TenPound</Description>
    <Description>Ten Pounds Off</Description>
    <Benefit>
      <FixedAmountDiscount>
        <Amount>10.00</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </FixedAmountDiscount>
    </Benefit>
  </Promotion>
</Promotions>
```
### CurrencyCode

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The type of currency for the WeightBased or the ItemBasedQuantity. Currently, only GBP is accepted.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if CalculatePromotions is “true” in the cart XML.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsResponse/ShippingMethods/ShippingMethod/Rate/WeightBased/CurrencyCode</code></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>3-character string</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td></td>
</tr>
</tbody>
</table>

```xml
<ShippingMethods>
  <ShippingMethod>
    <ShippingMethodId>UPSGround</ShippingMethodId>
    <ServiceLevel>Standard</ServiceLevel>
    <Rate>
      <WeightBased>
        <Amount>3.49</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </WeightBased>
      <Rate>
        <IncludedRegions>
          <USZipRegion>98104</USZipRegion>
        </IncludedRegions>
      </Rate>
    </Rate>
  </ShippingMethod>
</ShippingMethods>
```

### Description

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The description for the promotion specified in the document.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if CalculatePromotions is “true” in the cart XML.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsResponse/Promotions/Promotion/Description</code></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>40-character string</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td></td>
</tr>
</tbody>
</table>

```xml
<Promotions>
  <Promotion>
    <PromotionId>TenPound</PromotionId>
    <Description>Ten Pounds Off</Description>
    <Benefit>
      <FixedAmountDiscount>
        <Amount>10.00</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </FixedAmountDiscount>
    </Benefit>
  </Promotion>
</Promotions>
```
### DiscountRate

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The container for the percent-off promotion specified in the document.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if CalculatePromotions is “true” in the cart XML and the discount type is discount rate (percent off).</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsResponse/Promotions/Promotion/Benefit/FixedAmountDiscount</code></td>
</tr>
<tr>
<td><strong>Required subentry</strong></td>
<td>Amount</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>&lt;Promotions&gt;</code>&lt;Promotion&gt;&lt;PromotionId&gt;SixHalfPercent&lt;/Description&gt;&lt;Description&gt;6 and One Half Percent Off&lt;/Description&gt;&lt;Benefit&gt;&lt;DiscountRate&gt;&lt;Amount&gt;0.065&lt;/Amount&gt;&lt;/DiscountRate&gt;&lt;/Benefit&gt;&lt;/Promotion&gt;&lt;/Promotion&gt;`</td>
</tr>
</tbody>
</table>

### Error

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The container element for the error information in your response.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Either CallbackOrders or Error is required in a response.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/MerchantCalculationResponse/Response/Error</code></td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
<td>Code, Message</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
<tr>
<td><strong>ExcludedRegions</strong></td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>---</td>
</tr>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>The container for the regions excluded. Note that ExcludedRegions is evaluated first, and then IncludedRegions. Also, both ExcludedRegions and IncludedRegions can be set to the same value, which we evaluate to mean that you don’t ship to the buyer’s address.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>No</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsResponse/ShippingMethods/ExcludedRegions</td>
</tr>
<tr>
<td><strong>Subentries</strong></td>
<td>PredefinedRegion</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
</tbody>
</table>
| **Example**         | `<ShippingMethods>
    <ShippingMethod>
      <ShippingMethodId>UPSGround</ShippingMethodId>
      <ServiceLevel>Standard</ServiceLevel>
      <Rate>
        <WeightBased>
          <Amount>3.49</Amount>
          <CurrencyCode>GBP</CurrencyCode>
        </WeightBased>
        <Rate>
          <IncludedRegions>
            <USZipRegion>98104</USZipRegion>
          </IncludedRegions>
          <ExcludedRegions>
            <USZipRegion>98104</USZipRegion>
          </ExcludedRegions>
        </Rate>
      </Rate>
    </ShippingMethod>
  </ShippingMethods>` |
<table>
<thead>
<tr>
<th>FixedAmountDiscount</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
</tr>
<tr>
<td><strong>Definition</strong></td>
</tr>
<tr>
<td><strong>Required</strong></td>
</tr>
<tr>
<td><strong>XPath</strong></td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
</tr>
<tr>
<td><strong>Type</strong></td>
</tr>
</tbody>
</table>
| **Example**         | <Promotions>
|                     |   <Promotion>
|                     |     <PromotionId>TenPound</Description>
|                     |     <Description>Ten Pounds Off</Description>
|                     |     <Benefit>
|                     |       <FixedAmountDiscount>
|                     |         <Amount>10.00</Amount>
|                     |         <CurrencyCode>GBP</CurrencyCode>
|                     |       </FixedAmountDiscount>
|                     |     </Benefit>
|                     |   </Promotion>
|                     | </Promotions> |
# IncludedRegions

<table>
<thead>
<tr>
<th><strong>Use</strong></th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The container for the regions supported.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if <code>CalculatePromotions</code> is “true” in the cart XML.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsResponse/ShippingMethods/IncludedRegions</code></td>
</tr>
<tr>
<td><strong>Required Subentries</strong></td>
<td>PredefinedRegion</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>&lt;ShippingMethods&gt;</code>&lt;br&gt;<code>&lt;ShippingMethod&gt;</code>&lt;br&gt;<code>&lt;ShippingMethodId&gt;UPSGround&lt;/ShippingMethodId&gt;</code>&lt;br&gt;<code>&lt;ServiceLevel&gt;Standard&lt;/ServiceLevel&gt;</code>&lt;br&gt;<code>&lt;Rate&gt;</code>&lt;br&gt;<code>&lt;WeightBased&gt;</code>&lt;br&gt;<code>&lt;Amount&gt;3.49&lt;/Amount&gt;</code>&lt;br&gt;<code>&lt;CurrencyCode&gt;GBP&lt;/CurrencyCode&gt;</code>&lt;br&gt;<code>&lt;/WeightBased&gt;</code>&lt;br&gt;<code>&lt;Rate&gt;</code>&lt;br&gt;<code>&lt;IncludedRegions&gt;</code>&lt;br&gt;<code>&lt;USZipRegion&gt;98104&lt;/USZipRegion&gt;</code>&lt;br&gt;<code>&lt;/IncludedRegions&gt;</code>&lt;br&gt;<code>&lt;/ShippingMethod&gt;</code>&lt;br&gt;<code>&lt;/ShippingMethods&gt;</code></td>
</tr>
</tbody>
</table>

# ItemQuantityBased

<table>
<thead>
<tr>
<th><strong>Use</strong></th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>Container for the shipping charge calculation if item-based.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>WeightBased, ItemQuantityBased, or ShipmentBased is required.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsResponse/ShippingMethods/ShippingMethod/Rate/ItemQuantityBased</code></td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
<td>Amount, CurrencyCode</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>&lt;ShippingMethods&gt;</code>&lt;br&gt;<code>&lt;ShippingMethod&gt;</code>&lt;br&gt;<code>&lt;ShippingMethodId&gt;UPSGround&lt;/ShippingMethodId&gt;</code>&lt;br&gt;<code>&lt;ServiceLevel&gt;Standard&lt;/ServiceLevel&gt;</code>&lt;br&gt;<code>&lt;Rate&gt;</code>&lt;br&gt;<code>&lt;ItemQuantityBased&gt;</code>&lt;br&gt;<code>&lt;Amount&gt;3.49&lt;/Amount&gt;</code>&lt;br&gt;<code>&lt;CurrencyCode&gt;GBP&lt;/CurrencyCode&gt;</code>&lt;br&gt;<code>&lt;/ItemQuantityBased&gt;</code>&lt;br&gt;<code>&lt;Rate&gt;</code>&lt;br&gt;<code>&lt;IncludedRegions&gt;</code>&lt;br&gt;<code>&lt;USZipRegion&gt;98104&lt;/USZipRegion&gt;</code>&lt;br&gt;<code>&lt;/IncludedRegions&gt;</code>&lt;br&gt;<code>&lt;/ShippingMethod&gt;</code>&lt;br&gt;<code>&lt;/ShippingMethods&gt;</code></td>
</tr>
</tbody>
</table>
### MerchantCalculationRequest

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>The container for the request.</td>
</tr>
<tr>
<td>Required</td>
<td>Yes</td>
</tr>
<tr>
<td>XPath</td>
<td>/OrderCalculationsRequest</td>
</tr>
<tr>
<td>Type</td>
<td>Container</td>
</tr>
</tbody>
</table>
| Example   | ```
<?xml version="1.0" encoding="UTF-8"?>
<OrderCalculationsRequest>
  ...
</MerchantCalculationRequest>
``` |

### Message

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>A reader-friendly explanation of the error.</td>
</tr>
<tr>
<td>Required</td>
<td>Either CallbackOrders or Error is required in a response.</td>
</tr>
<tr>
<td>XPath</td>
<td>/MerchantCalculationResponse/Response/Error/Message</td>
</tr>
<tr>
<td>Type</td>
<td>String</td>
</tr>
<tr>
<td>Value/Range</td>
<td>40-character string</td>
</tr>
</tbody>
</table>
| Example   | ```
<Response>
  <Error>
    <Code>INTERNAL_SERVER_ERROR</Code>
    <Message>Under Maintenance</Message>
  </Error>
</Response>
``` |
<table>
<thead>
<tr>
<th><strong>OrderCalculationCallbacks</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>The container for all other entries regarding callbacks. If this entry is missing, no callbacks are performed.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if you want to supply your own calculated amounts for shipping or promotional discounts.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationCallbacks</td>
</tr>
<tr>
<td><strong>Required subentry</strong></td>
<td>OrderCallbackEndpoint</td>
</tr>
<tr>
<td><strong>Optional subentries</strong></td>
<td>CalculatePromotions, CalculateShippingRates, ProcessOrderOnCallbackFailure</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td></td>
</tr>
</tbody>
</table>
```xml
<OrderCalculationCallbacks>
  ...
  <OrderCallbackEndpoint>
    https://my.endpoint.com/receive.php
  </OrderCallbackEndpoint>
  ...
</OrderCalculationCallbacks>
``` |

<table>
<thead>
<tr>
<th><strong>OrderCalculationsRequest</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>The container for the request.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsRequest</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td></td>
</tr>
</tbody>
</table>
```xml
<?xml version="1.0" encoding="UTF-8" ?>
  ...
</OrderCalculationsRequest>
``` |

<table>
<thead>
<tr>
<th><strong>OrderCalculationsResponse</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>The main container for all response information. Must use the XMLNS <a href="http://payments.amazon.com/checkout/2009-05-15/">http://payments.amazon.com/checkout/2009-05-15/</a> or later.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsResponse</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td></td>
</tr>
</tbody>
</table>
```xml
  ...
</OrderCalculationsResponse>
``` |
### OrderCallbackEndpoint

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The merchant URL that the callback service will send the callback request to. Calls made to the Production endpoint are made only over HTTPS, using port 443. Calls made to the Sandbox endpoint are made over HTTP, using port 80.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationCallbacks/OrderCallbackEndpoint</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>URL; uses HTTPS for Production and HTTP for Sandbox.</td>
</tr>
</tbody>
</table>
| **Example** | `<OrderCalculationCallbacks>  
  <CalculatePromotions>true</CalculatePromotions>  
  <CalculateShippingRates>true</CalculateShippingRates>  
  <OrderCallbackEndpoint>https://my.endpoint.com/receive.php</OrderCallbackEndpoint>  
  <ProcessOrderOnCallbackFailure>true</ProcessOrderOnCallbackFailure>  
</OrderCalculationCallbacks>` |

### PostalCode

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The postal code for the buyer where the item will be shipped to.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsRequest/CallbackOrders/CallbackOrder/Address/AddressId/PostalCode</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>40-character string</td>
</tr>
</tbody>
</table>
| **Example** | `<Address>  
  <AddressId>AddressId</AddressId>  
  <AddressFieldOne>1200 5th Avenue South</AddressFieldOne>  
  <AddressFieldTwo>Suite 116</AddressFieldTwo>  
  <AddressFieldThree>Station M162K7</AddressFieldThree>  
  <City>Seattle</City>  
  <State>WA</State>  
  <PostalCode>98104</PostalCode>  
</Address>` |
### ProcessOrderOnCallbackFailure

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>Also called the fallback flag; stipulates how Amazon Payments will handle the order if the callback fails. If “true”, order processing continues using overrides specified in the original cart XML or using Seller Central settings. If “false”, order processing halts and the buyer will see an error page. If you do not specify this entry, the value is set to “true”.</td>
</tr>
<tr>
<td>Required</td>
<td>No</td>
</tr>
<tr>
<td>XPath</td>
<td>/OrderCalculationCallbacks/ProcessOrderOnCallbackFailure</td>
</tr>
<tr>
<td>Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Value/Range</td>
<td>true</td>
</tr>
</tbody>
</table>

**Example**

```xml
<OrderCalculationCallbacks>
  <CalculatePromotions>true</CalculatePromotions>
  <CalculateShippingRates>true</CalculateShippingRates>
  <OrderCallbackEndpoint>
    https://my.endpoint.com/receive.php
  </OrderCallbackEndpoint>
  <ProcessOrderOnCallbackFailure>true</ProcessOrderOnCallbackFailure>
</OrderCalculationCallbacks>
```

### Promotion

<table>
<thead>
<tr>
<th>Use</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>The container for a promotion.</td>
</tr>
<tr>
<td>Required</td>
<td>Yes, if CalculatePromotions is “true” in the cart XML.</td>
</tr>
<tr>
<td>XPath</td>
<td>/OrderCalculationsResponse/Promotions/Promotion</td>
</tr>
<tr>
<td>Required subentry</td>
<td>PromotionId, Description, Benefit</td>
</tr>
<tr>
<td>Type</td>
<td>Container</td>
</tr>
</tbody>
</table>

**Example**

```xml
<Promotions>
  <Promotion>
    <PromotionId>TenPound</PromotionId>
    <Description>Ten Pounds Off</Description>
    <Benefit>
      <FixedAmountDiscount>
        <Amount>10.00</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </FixedAmountDiscount>
    </Benefit>
  </Promotion>
</Promotions>
```
<table>
<thead>
<tr>
<th><strong>PromotionId</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
</tr>
<tr>
<td><strong>Definition</strong></td>
</tr>
<tr>
<td><strong>Required</strong></td>
</tr>
<tr>
<td><strong>XPath</strong></td>
</tr>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
</tr>
</tbody>
</table>
| **Example** | `<CallbackOrders>  
  <CallbackOrder>  
    <Address>  
      <AddressId>CustomerAddress1</AddressId>  
    </Address>  
    <CallbackOrderItems>  
      <CallbackOrderItem>  
        <SKU>JKL909</SKU>  
        <PromotionIds>  
          <PromotionId>TenPound</PromotionId>  
        </PromotionIds>  
        <ShippingMethodIds>  
          <ShippingMethodId>US Standard</ShippingMethodId>  
          <ShippingMethodId>US NextDay</ShippingMethodId>  
        </ShippingMethodIds>  
      </CallbackOrderItem>  
    </CallbackOrderItems>  
  </CallbackOrder>  
  ...  
  <Promotions>  
    <Promotion>  
      <PromotionId>TenPound</PromotionId>  
      <Description>Ten Pounds Off</Description>  
      <Benefit>  
        <FixedAmountDiscount>  
          <Amount>10.00</Amount>  
          <CurrencyCode>GBP</CurrencyCode>  
        </FixedAmountDiscount>  
      </Benefit>  
      ...  
    </Promotion>  
  </Promotions>  
</CallbackOrders>` |
### PromotionIds

**Use** XML

**Definition** A container for applicable promotions for the shipping item.

**Required** Yes, if `CalculatePromotions` is “true” in the cart XML.

**XPath** /OrderCalculationsResponse/CallbackOrders/CallbackOrder/CallbackOrderItems/CallbackOrderItem/PromotionIds

**Required subentry** PromotionId

**Type** Container

**Example**

```xml
<CallbackOrders>
    <CallbackOrder>
        <Address>
            <AddressId>CustomerAddress1</AddressId>
        </Address>
        <CallbackOrderItems>
            <CallbackOrderItem>
                <SKU>JKL909</SKU>
                <PromotionIds>
                    <PromotionId>TenPound</PromotionId>
                </PromotionIds>
                <ShippingMethodIds>
                    <ShippingMethodId>US Standard</ShippingMethodId>
                    <ShippingMethodId>US NextDay</ShippingMethodId>
                </ShippingMethodIds>
            </CallbackOrderItem>
        </CallbackOrderItems>
    </CallbackOrder>
</CallbackOrders>
```

### Promotions

**Use** XML

**Definition** The container for the promotion rules.

**Required** Yes, if `CalculatePromotions` is “true” in the cart XML.

**XPath** /OrderCalculationsResponse/Promotions

**Required subentry** Promotion

**Type** Container

**Example**

```xml
<Promotions>
    <Promotion>
        <PromotionId>TenPound</PromotionId>
        <Description>Ten Pounds Off</Description>
        <Benefit>
            <FixedAmountDiscount>
                <Amount>10.00</Amount>
                <CurrencyCode>GBP</CurrencyCode>
            </FixedAmountDiscount>
        </Benefit>
    </Promotion>
</Promotions>
```
### Rate

<table>
<thead>
<tr>
<th><strong>Use</strong></th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>Container for the type of shipping calculation.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if CalculateShippingRates is “true” in the cart XML.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsResponse/ShippingMethods/Rate</code></td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
<td>WeightBased</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
</tbody>
</table>
| **Example** | `<ShippingMethods>
    <ShippingMethod>
        <ShippingMethodId>UPSGround</ShippingMethodId>
        <ServiceLevel>Standard</ServiceLevel>
        <Rate>
            <WeightBased>
                <Amount>3.49</Amount>
                <CurrencyCode>GBP</CurrencyCode>
            </WeightBased>
            < IncludedRegions>
                <USZipRegion>98104</USZipRegion>
            </IncludedRegions>
        </Rate>
    </ShippingMethod>
</ShippingMethods>` |

### Response

<table>
<thead>
<tr>
<th><strong>Use</strong></th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>The container element for all your responses.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsResponse/Response/</code></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>CallbackOrders</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>&lt;Response&gt;...&lt;/Response&gt;</code></td>
</tr>
</tbody>
</table>
**ServiceLevel**

**Use** XML

**Definition** The name for the level of service you are defining in your shipping method.

**Required** Yes, if `CalculateShippingRates` is “true” in the cart XML.

**XPath** `/OrderCalculationsResponse/ShippingMethods/ServiceLevel`

**Type** Enumerated string

**Value/Range** OneDay | TwoDay | Standard | Expedited

**Example**

```
<ShippingMethods>
  <ShippingMethod>
    <ShippingMethodId>UPSGround</ShippingMethodId>
    <ServiceLevel>Standard</ServiceLevel>
    <Rate>
      <WeightBased>
        <Amount>3.49</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </WeightBased>
      <Rate>
        <IncludedRegions>
          <USZipRegion>98104</USZipRegion>
        </IncludedRegions>
      </Rate>
    </Rate>
  </ShippingMethod>
</ShippingMethods>
```

**ShipmentBased**

**Use** XML

**Definition** Container for the shipping charge calculation if item-based.

**Required** `WeightBased`, `ItemQuantityBased`, or `ShipmentBased` is required.

**XPath** `/OrderCalculationsResponse/ShippingMethods/ShippingMethod/Rate/ShipmentBased`

**Required subentries** Amount, CurrencyCode

**Type** Container

**Example**

```
<ShippingMethods>
  <ShippingMethod>
    <ShippingMethodId>UPSGround</ShippingMethodId>
    <ServiceLevel>Standard</ServiceLevel>
    <Rate>
      <ShipmentBased>
        <Amount>7.99</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </ShipmentBased>
      <Rate>
        <IncludedRegions>
          <USZipRegion>98104</USZipRegion>
        </IncludedRegions>
      </Rate>
    </Rate>
  </ShippingMethod>
</ShippingMethods>
```
<table>
<thead>
<tr>
<th><strong>ShippingMethod</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
</tr>
<tr>
<td><strong>Definition</strong></td>
</tr>
<tr>
<td><strong>Required</strong></td>
</tr>
<tr>
<td><strong>XPath</strong></td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
</tr>
<tr>
<td><strong>Optional subentry</strong></td>
</tr>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td><strong>Example</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>ShippingMethodId</strong></td>
</tr>
<tr>
<td>----------------------</td>
</tr>
<tr>
<td><strong>Use</strong></td>
</tr>
<tr>
<td><strong>Definition</strong></td>
</tr>
<tr>
<td><strong>Required</strong></td>
</tr>
<tr>
<td><strong>XPath</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
</tr>
</tbody>
</table>
| **Example**          | `<CallbackOrders>`  
|                      | `  <CallbackOrder>`  
|                      | `    <Address>`  
|                      | `       <AddressId>CustomerAddress1</AddressId>`  
|                      | `    </Address>`  
|                      | `  <CallbackOrderItems>`  
|                      | `    <CallbackOrderItem>`  
|                      | `       <SKU>JKL909</SKU>`  
|                      | `       <PromotionIds>`  
|                      | `          <PromotionId>TenPound</PromotionId>`  
|                      | `       </PromotionIds>`  
|                      | `    <ShippingMethodIds>`  
|                      | `       <ShippingMethodId>US Standard</ShippingMethodId>`  
|                      | `    </ShippingMethodIds>`  
|                      | `  </CallbackOrderItem>`  
|                      | `</CallbackOrderItems>`  
|                      | `</CallbackOrder>`  
|                      | `</CallbackOrders>`  
|                      | `...`  
|                      | `<ShippingMethods>`  
|                      | `    <ShippingMethod>`  
|                      | `       <ShippingMethodId>US Standard</ShippingMethodId>`  
|                      | `       <ServiceLevel>Standard</ServiceLevel>`  
|                      | `       <Rate>`  
|                      | `          <WeightBased>`  
|                      | `             <Amount>3.49</Amount>`  
|                      | `             <CurrencyCode>GBP</CurrencyCode>`  
|                      | `        </WeightBased>`  
|                      | `       </Rate>`  
|                      | `    </IncludedRegions>`  
|                      | `       <USZipRegion>98104</USZipRegion>`  
|                      | `    </IncludedRegions>`  
|                      | `</ShippingMethod>`  
<p>|                      | <code>&lt;/ShippingMethods&gt;</code> |</p>
<table>
<thead>
<tr>
<th><strong>ShippingMethodIds</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>A container for the shipping methods.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if CalculateShippingRates is “true” in the cart XML.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsResponse/CallbackOrderItems/CallbackOrderItem/ShippingMethodIds</td>
</tr>
<tr>
<td><strong>Required subentry</strong></td>
<td>ShippingMethodId</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
</tbody>
</table>
| **Example**           | `<CallbackOrders>
  <CallbackOrder>
    <Address>
      <AddressId>CustomerAddress1</AddressId>
    </Address>
    <CallbackOrderItems>
      <CallbackOrderItem>
        <SKU>JKL909</SKU>
        <PromotionIds>
          <PromotionId>TenPound</PromotionId>
        </PromotionIds>
        <ShippingMethodIds>
          <ShippingMethodId>US Standard</ShippingMethodId>
          <ShippingMethodId>US NextDay</ShippingMethodId>
        </ShippingMethodIds>
      </CallbackOrderItem>
    </CallbackOrderItems>
  </CallbackOrder>
</CallbackOrders>` |
<table>
<thead>
<tr>
<th><strong>ShippingMethods</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>The container for the shipping methods.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, if CalculateShippingRates is “true” in the cart XML.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsResponse/ShippingMethods</td>
</tr>
<tr>
<td><strong>Required subentry</strong></td>
<td>ShippingMethod</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
</tbody>
</table>

**Example**

```xml
<ShippingMethods>
  <ShippingMethod>
    <ShippingMethodId>UPSGround</ShippingMethodId>
    <ServiceLevel>Standard</ServiceLevel>
    <Rate>
      <WeightBased>
        <Amount>3.49</Amount>
        <CurrencyCode>GBP</CurrencyCode>
      </WeightBased>
      <Rate>
        <IncludedRegions>
          <USZipRegion>98104</USZipRegion>
        </IncludedRegions>
      </Rate>
    </Rate>
  </ShippingMethod>
</ShippingMethods>
```
<table>
<thead>
<tr>
<th><strong>SKU</strong></th>
<th><strong>XML</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>The merchant-defined identification code (Stock-keeping Unit), a unique identifier for the CallbackOrderItem or Item.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes, for both the cart XML and the callback response.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/Order/Cart/Items/Item/SKU&lt;br/&gt;/OrderCalculationsResponse/CallbackOrderItems/CallbackOrderItem/SKU</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>40-character string</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>&lt;Cart&gt;&lt;Items&gt;&lt;Item&gt;&lt;SKU&gt;JKL909&lt;/SKU&gt;&lt;MerchantId&gt;AEIOU1234AEIOU&lt;/MerchantId&gt;&lt;Title&gt;Calvin and Hobbes Reliquary&lt;/Title&gt;&lt;Price&gt;&lt;Amount&gt;29.99&lt;/Amount&gt;&lt;CurrencyCode&gt;GBP&lt;/CurrencyCode&gt;&lt;/Price&gt;&lt;Quantity&gt;1&lt;/Quantity&gt;&lt;/Item&gt;&lt;/Items&gt;&lt;/Cart&gt;&lt;OR&gt;&lt;CallbackOrderItem&gt;&lt;SKU&gt;JKL909&lt;/SKU&gt;&lt;/CallbackOrderItem&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>State</strong></th>
<th><strong>XML</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>The 2-letter abbreviation of the buyer's state where the Item will be shipped to.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td>/OrderCalculationsRequest/CallbackOrders/CallbackOrder/Address/AddressId/State</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Value/Range</strong></td>
<td>2-character string</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>&lt;Address&gt;&lt;AddressId&gt;AddressId&lt;/AddressId&gt;&lt;AddressFieldOne&gt;1200 5th Avenue South&lt;/AddressFieldOne&gt;&lt;AddressFieldTwo&gt;Suite 116&lt;/AddressFieldTwo&gt;&lt;AddressFieldThree&gt;Station M162K7&lt;/AddressFieldThree&gt;&lt;City&gt;Seattle&lt;/City&gt;&lt;State&gt;WA&lt;/State&gt;&lt;PostalCode&gt;98104&lt;/PostalCode&gt;&lt;CountryCode&gt;US&lt;/CountryCode&gt;&lt;/Address&gt;</td>
</tr>
<tr>
<td><strong>WeightBased</strong></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>---</td>
</tr>
<tr>
<td><strong>Use</strong></td>
<td>XML</td>
</tr>
<tr>
<td><strong>Definition</strong></td>
<td>Container for the shipping charge calculation if weight-based.</td>
</tr>
<tr>
<td><strong>Required</strong></td>
<td>WeightBased, ItemQuantityBased, or ShipmentBased is required.</td>
</tr>
<tr>
<td><strong>XPath</strong></td>
<td><code>/OrderCalculationsResponse/ShippingMethods/ShippingMethod/Rate/WeightBased</code></td>
</tr>
<tr>
<td><strong>Required subentries</strong></td>
<td>Amount, CurrencyCode</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Container</td>
</tr>
</tbody>
</table>
| **Example**     | `<ShippingMethods>`  
|                  |   `<ShippingMethod>`  
|                  |     `<ShippingMethodId>UPSGround</ShippingMethodId>`  
|                  |     `<ServiceLevel>Standard</ServiceLevel>`  
|                  |     `<Rate>`  
|                  |       `<WeightBased>`  
|                  |         `<Amount>3.49</Amount>`  
|                  |         `<CurrencyCode>GBP</CurrencyCode>`  
|                  |       `</WeightBased>`  
|                  |     `<Rate>`  
|                  |     `<IncludedRegions>`  
|                  |       `<USZipRegion>98104</USZipRegion>`  
|                  |     `</IncludedRegions>`  
|                  |   `</ShippingMethod>`  
|                  | `</ShippingMethods>` |
Appendix B: The Callback Schema

You can download the callback.xsd from our servers. Alternately, you can view the content below. For your reference, you can also download the order.xsd from our servers.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.w3.org/2001/XMLSchema"
        http://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified">
    <xs:include schemaLocation="order.xsd"/>
    <xs:element name="OrderCalculationsRequest" type="tns:OrderCalculationsRequest" />
    <xs:element name="OrderCalculationsResponse" type="tns:OrderCalculationsResponse" />
    <xs:element name="OrderCalculationsError" type="tns:OrderCalculationsError" />

    <xs:complexType name="OrderCalculationsRequest">
        <xs:sequence>
            <xs:element name="CallbackReferenceId" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="OrderCalculationCallbacks" type="tns:OrderCalculationCallbacks" minOccurs="1" maxOccurs="1" />
            <xs:element name="ClientRequestId" type="xs:string" minOccurs="0"/>
            <xs:element name="IntegratorId" type="xs:string" minOccurs="0"/>
            <xs:element name="IntegratorName" type="xs:string" minOccurs="0"/>
            <xs:element name="Cart" type="tns:Cart" maxOccurs="1" />
            <xs:element name="CallbackOrders" type="tns:CallbackOrders" minOccurs="1" maxOccurs="1" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="OrderCalculationsResponse">
        <xs:sequence>
            <xs:element name="Response">
                <xs:complexType>
                    <xs:sequence>
                        <xs:choice>
                            <xs:element name="Error" type="tns:Error" maxOccurs="1" />
                            <xs:element name="CallbackOrders" type="tns:CallbackOrders" minOccurs="1" maxOccurs="1" />
                        </xs:choice>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="TaxTables" type="tns:TaxTables" minOccurs="0" />
            <xs:element name="Promotions" type="tns:Promotions" minOccurs="0" />
            <xs:element name="ShippingMethods" type="tns:ShippingMethods" minOccurs="0" />
            <xs:element name="CartPromotionId" type="xs:string" minOccurs="0" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="OrderCalculationsError">
        <xs:sequence>
            <xs:element name="OrderCalculationsErrorCode" type="xs:string" minOccurs="1" maxOccurs="1" />
            <xs:element name="OrderCalculationsErrorMessage" type="xs:string" minOccurs="1" maxOccurs="1" />
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```
<xs:element name="OrderCalculationsRequest"
type="tns:OrderCalculationsRequest" minOccurs="1" maxOccurs="1"/>
<xs:element name="OrderCalculationsResponse" type="xs:string"
minOccurs="1" maxOccurs="1"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="Address">
<xs:sequence>
<xs:element name="AddressId" type="xs:string" minOccurs="1"
maxOccurs="1"/>
<xs:element name="AddressFieldOne" type="xs:string" minOccurs="0"
maxOccurs="1"/>
<xs:element name="AddressFieldTwo" type="xs:string" minOccurs="0"
maxOccurs="1"/>
<xs:element name="AddressFieldThree" type="xs:string" minOccurs="0"
maxOccurs="1"/>
<xs:element name="City" type="xs:string" minOccurs="0"
maxOccurs="1"/>
<xs:element name="State" type="xs:string" minOccurs="0"
maxOccurs="1"/>
<xs:element name="PostalCode" type="xs:string" minOccurs="0"
maxOccurs="1"/>
<xs:element name="CountryCode" type="xs:string" minOccurs="0"
maxOccurs="1"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="CallbackOrders">
<xs:sequence>
<xs:element name="CallbackOrder" type="tns:CallbackOrder" minOccurs="1"
maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="CallbackOrder">
<xs:sequence>
<xs:element name="Address" type="tns:Address" minOccurs="1"
maxOccurs="1"/>
<xs:element name="CallbackOrderItems" type="tns:CallbackOrderItems"
minOccurs="1" maxOccurs="1"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="CallbackOrderItems">
<xs:sequence>
<xs:element name="CallbackOrderItem" type="tns:CallbackOrderItem" minOccurs="1"
maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="CallbackOrderItem">
<xs:sequence>
<xs:element name="SKU" type="xs:string" minOccurs="1" maxOccurs="1"/>
<xs:element name="TaxTableId" type="xs:string" minOccurs="0"/>
<xs:element name="PromotionIds" type="tns:PromotionIds" minOccurs="0"/>
<xs:element name="ShippingMethodIds" type="tns:ShippingMethodIds"
minOccurs="0"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="Error">
<xs:sequence>
<xs:element name="Code" type="tns:ErrorCode"/>
<xs:element name="Message" type="xs:string"/>
</xs:sequence>
</xs:complexType>
<xs:sequence>
  </xs:complexType>

  <xs:simpleType name="ErrorCode">
    <xs:restriction base="xs:string">
      <xs:enumeration value="INVALID_SHIPPING_ADDRESS"/>
      <xs:enumeration value="INVALID_REQUEST_ERROR"/>
      <xs:enumeration value="INTERNAL_SERVER_ERROR"/>
      <xs:enumeration value="SERVICE_UNAVAILABLE"/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
Appendix C: Accepted SSL Certificates

Checkout by Amazon currently accepts SSL certificates with root certificates from any of the following Certificate Authorities (CA):

- ABAecom (sub., Am. Bankers Assn.) Root CA
- AddTrust External Root
- AddTrust Low-Value Services Root
- AddTrust Public Services Root
- AddTrust Qualified Certificates Root
- America Online Root Certification Authority 1
- America Online Root Certification Authority 2
- AOL Time Warner Root Certification Authority 1
- AOL Time Warner Root Certification Authority 2
- Baltimore CyberTrust Root
- beTRUSTed Root CA
- beTRUSTed Root CA - Entrust Implementation
- beTRUSTed Root CA - RSA Implementation
- beTRUSTed Root CA-Baltimore Implementation
- Camerfirma Chambers of Commerce Root
- Camerfirma Global Chambersign Root
- Certplus Class 2 Primary CA
- Certum Root CA
- Comodo AAA Services root
- COMODO Certification Authority
- Comodo Secure Services root
- Comodo Trusted Services root
- DigiCert Assured ID Root CA
- DigiCert Global Root CA
- DigiCert High Assurance EV Root CA
- Digital Signature Trust Co. Global CA 1
- Digital Signature Trust Co. Global CA 2
- Digital Signature Trust Co. Global CA 3
- Digital Signature Trust Co. Global CA 4
- DST ACES CA X6
- DST Root CA X3
- Entrust Root Certification Authority
- Entrust.net Global Secure Personal CA
- Entrust.net Global Secure Server CA
- Entrust.net Premium 2048 Secure Server CA
- Entrust.net Secure Personal CA
- Entrust.net Secure Server CA
- Equifax Secure CA
- Equifax Secure eBusiness CA 1
- Equifax Secure eBusiness CA 2
- Equifax Secure Global eBusiness CA
- Firmaprofesional Root CA
- GeoTrust Extended Validation SSL CA
- GeoTrust Global CA
- GeoTrust Global CA 2
- GeoTrust Primary Certification Authority
- GeoTrust Universal CA
- GeoTrust Universal CA 2
- GlobalSign Root CA
- GlobalSign Root CA - R2
- GoDaddy Class 2 CA
- GTE CyberTrust Global Root
- GTE CyberTrust Root CA
- IPS Chained CAs root
- IPS CLASE1 root
- IPS CLASE3 root
- IPS CLASEA1 root
- IPS CLASEA3 root
- IPS Servidores root
- IPS Timestamping root
- NetLock Business (Class B) Root
- NetLock Express (Class C) Root
- NetLock Notary (Class A) Root
- NetLock Qualified (Class QA) Root
- Network Solutions Certificate Authority
- QuoVadis Root CA
- QuoVadis Root CA 2
- QuoVadis Root CA 3
- RSA Root Certificate 1
- RSA Security 2048 v3
11.1 Security Certificates and Transactions with Amazon Payments

We check the bottom of the certificate chain (usually a domain such as mysite.com) and will not establish an SSL Handshake connection with a site using a Certificate Signature Algorithm that uses MD5 with RSA Encryption.

To examine your site’s encryption algorithm, follow these steps
11.1.1 In Firefox 3.x

1. Go to your site using the HTTPS:// secure protocol.
2. Click the site icon to the left of the domain name (or click Tools > Page Info).

This brings up an information dialog box about the host.

![Information dialog box](image.png)
3. Click the **More Information** button to display the Page Info dialog box.

4. On the Page Info box, click the **Security** icon, and then click the **View Certificate** button to display the Certificate Viewer dialog box.
5. On the Certificate Viewer box, click the **Details** tab; then, under the **Certificate Fields** list box, scroll down and click **Certificate Signature Algorithm** to display the **Field Value**. The **Field Value** box displays the **certificate algorithm used**.

![Certificate Viewer](image)

6. If the **Field Value** shows “MD5 With RSA Encryption,” the certificate is not valid for use with Amazon Payments transactions.
11.1.2 In IE 7.x

1. Go to your site using the HTTPS:// secure protocol.
2. Click the security icon (a lock) to the right of the domain name (or click Page Info > Security Report).

This brings up the Website Identification pop-up window.
3. On the Website Identification window, click **View Certificates** to display the **Certificate** dialog box.

![Certificate dialog box]

- **Certificate Information**
  - This certificate is intended for the following purpose(s):
    - Ensures the identity of a remote computer
  - Issued to: secure.wndjii.com
  - Issued by: Equifax Secure Global eBusiness CA-1
  - Valid from 10/29/2007 to 11/27/2012
4. On the Certificate box, click the **Details** tab to display the Signature algorithm used.

5. If this value shows “md5RSA,” the certificate is not valid for use with Amazon Payments transactions.