



BE PAYMENT READY

PHP - Moneris Kount Technical Specifications Guide - Kount Risk Inquiries API Merchant Integration Guide

Version: 1.0.2

Copyright © Moneris Solutions, 2017

All rights reserved. No part of this publication may be reproduced, stored in retrieval systems, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Moneris Solutions Corporation.

Table of Contents

| | |
|---|-----------|
| Table of Contents | 2 |
| 1 Understanding the Moneris Kount Services Process | 3 |
| 2 Kount Data Collector Process | 4 |
| 2.1 About the Kount Data Collector Process | 4 |
| 2.2 Data Collector Process Flow - Kount | 4 |
| 2.3 Data Collector Process Requirements | 4 |
| 2.4 Code Example for HTML iframe | 5 |
| 2.5 Code Example for Creating Session ID | 5 |
| 3 Kount Risk Inquiry Service (RIS) Process | 6 |
| 3.1 About the Kount Risk Inquiry Service Process | 6 |
| 3.2 Risk Inquiry Service Process Flows - Kount | 6 |
| 3.2.1 Risk Inquiry Service Process Flow - Prepayment | 6 |
| 3.2.2 Risk Inquiry Service Process Flow - Postpayment | 6 |
| 3.3 Transaction Types for Kount Risk Inquiries | 7 |
| 3.3.1 Kount Inquiry | 7 |
| 3.3.2 Kount Update | 12 |
| 3.4 Kount Risk Inquiry Responses | 15 |
| 3.4.1 The Kount Risk Score | 15 |
| 3.4.2 Auto Decision Response Codes | 15 |
| 4 Testing Your Moneris Kount Services Solution | 16 |
| 4.1 Getting Unique Test Store ID and API Token | 16 |
| 5 Moving to Production | 17 |
| 5.1 Activating a Store for Production | 17 |
| 5.2 Configuring a Store for Production | 17 |
| 6 Getting Help | 18 |
| Appendix A Definition of Request Fields | 19 |
| Appendix B Definitions of Response Fields | 26 |
| Appendix C Error Messages | 31 |
| Appendix D Response Codes for Kount | 33 |
| Appendix E Security Requirements | 34 |

1 Understanding the Moneris Kount Services Process

In a Kount risk inquiry, there are two sets of information collected in distinct processes that are then used to determine a transaction's risk score:

1. **Data Collector process:** This is where Kount gathers technical information about the customer's device, software and IP address. This process is done directly between your merchant website and Kount servers, and there is no involvement of the Moneris Gateway. For more detailed information about the Data Collector process, see Kount's website at www.kount.com.
2. **Risk Inquiry Service process:** This process collects and analyzes customer information as factors in the Kount risk score, such as the customer's card information, shipping and billing addresses, contact details etc. The Moneris Gateway integrates with the Kount Risk Inquiry Service process.

Although the Data Collector process does not involve the Moneris Gateway, the same session ID you generate to use in that process for a given transaction must be used in the subsequent Risk Inquiry Service process for that transaction.

2 Kount Data Collector Process

2.1 About the Kount Data Collector Process

The Data Collector process provides data related to the device initiating the transaction, and is transparent to the customer and sub second. The data collected is used in conjunction with the RIS data.

2.2 Data Collector Process Flow - Kount

The following sequence describes the flow of the Data Collector process:

1. Customer browses to merchant order form page containing Data Collector code
2. Customer browser automatically requests redirect on the Data Collector element
3. Customer browser is redirected to Kount servers
4. Kount collects device attributes
5. Kount directs customer browser to display static image hosted by merchant

2.3 Data Collector Process Requirements

1. Port 443 must be available to post and receive data from Kount
2. The following code must be added to your checkout process:
 - HTML iframe.

The iframe should be placed on the order form page where the payment method or credit card information is requested, usually near the bottom of the page.
 - Server side code:

The server side code consists of a logo.htm and logo.gif server side executable scripts.
The path to the server side code must be a fully qualified path.
 - Code to create a unique session identifier:

When the form page is accessed, a session identifier must be created and stored to be used as the session identifier in the subsequent RIS post from the merchant to Kount. This identifier must be unique for at least 30 days and must be unique for every transaction submitted by each unique customer. If a single session ID were to be used on multiple transactions, those transactions would link together and erroneously affect the persona information and Kount score.
3. A static image URL must be supplied to Kount:
 - The static image can be an image that is currently being displayed on the page or Kount can provide an image if desired

- If the path or image requires change by the merchant subsequent to moving into production, Kount must be notified of the new path or filename to avoid possible failure.

2.4 Code Example for HTML iframe

NOTE: The iframe has a minimum width=1 and height=1

Sample HTML iframe - Kount Data Collector Process

```
<iframe width=1 height=1 frameborder=0 scrolling=no src=https://MERCHANT_URL/logo.htm?
m=merchantId&s=sessionId>
<img width=1 height=1 src=https://MERCHANT_URL/logo.gif?m=merchantId&s=sessionId>
</iframe>
```

2.5 Code Example for Creating Session ID

Sample Creating a Unique Session Identifier

```
<?php
$sess = session_id();
if (!$sess) {
    // If the session hasn't already been started, start it now and look up the id
    session_start();
    $sess = session_id();
}
// The session id is now available for use in the variable $sess
// For more details and examples on working with sessions in PHP, see:
// http://us2.php.net/manual/en/book.session.php
// http://us2.php.net/session_start
// http://us2.php.net/session_id
?>
```

3 Kount Risk Inquiry Service (RIS) Process

3.1 About the Kount Risk Inquiry Service Process

The Risk Inquiry Service (RIS) joins device data provided from the data collector process with the customer order data sent from the merchant. Once the device data and the order data are combined, RIS evaluates and scores each transaction.

3.2 Risk Inquiry Service Process Flows - Kount

The process flow in the Kount Risk Inquiry Service Process depends on when the risk inquiry request is made in relation to authorizing the transaction on Moneris Gateway:

- RIS inquiry is sent **prior to attempting to authorize payment**
- RIS inquiry is sent **following the payment authorization request**

3.2.1 Risk Inquiry Service Process Flow - Prepayment

Performing a RIS inquiry (**kount_inquiry**) request before attempting an authorization from Moneris Gateway, below are the considerations regarding pre-authorization:

- All credit card information can be sent to Kount
- A RIS update (**kount_update**) call must be made to update order number and status of payment authorization including **payment_response**, **avs_response**, and **cvd_response** data provided by the Moneris Gateway.

For pre-payment, RIS inquiry requests set the following required fields as:

payment_response = A

3.2.2 Risk Inquiry Service Process Flow - Postpayment

Performing a RIS inquiry (**kount_inquiry**) request after Moneris Gateway has been contacted, below are the considerations regarding post-authorization:

- All payment gateway information can be passed to Kount (**payment_response**, **avs_response**, **cvd_response**) allowing rules to be created regarding AVS and CVV data.
- Single RIS query, no update is necessary.

3.3 Transaction Types for Kount Risk Inquiries

There are two transaction types designed for Kount Risk Inquiries:

Kount Inquiry

A risk inquiry request—can be performed either before or after performing a customer payment transaction. If performed before, the risk score will be less complete as it will not include most customer information.

Kount Update

Request to update the risk inquiry information, such as order number and payment authorization details.

3.3.1 Kount Inquiry

Kount Inquiry transaction object definition

```
$txnArray = array('type'=>'kount_inquiry', ...);
```

```
$mpgTxn = new mpgTransaction($txnArray);
```

HttpPostRequest object for Kount Inquiry transaction

```
$mpgRequest = new mpgRequest($mpgTxn);
```

```
$mpgHttpPost = new mpgHttpPost($store_id,$api_token,$mpgRequest);
```

Kount Inquiry transaction object request values

Table 1: Kount Inquiry transaction object mandatory values

| Value | Type | Limits | Set Method |
|-------------------|--------|----------------------------|--|
| Order ID | String | 50-character alpha-numeric | 'order_id'=>\$order_id |
| Kount merchant ID | String | 6-character alpha-numeric | 'kount_merchant_id'=>\$kount_merchant_id |
| Kount API key | String | Varies | 'kount_api_key'=>kount_api_key |
| Payment response | String | 1-character alpha-numeric | 'payment_response'=>\$payment_response |
| Payment token | String | 32-character alpha-numeric | 'payment_token'=>\$payment_token |
| Payment type | String | 4-character alpha-numeric | 'payment_type'=>\$payment_type |

| Value | Type | Limits | Set Method |
|--------------------------------------|--------|---|---|
| Currency | String | 3-character alpha-numeric | 'currency'=>\$currency |
| Call Center indicator | String | 1-character alpha-numeric | 'call_center_ind'=>\$call_center_ind |
| Session ID | String | 32-character alpha-numeric | 'session_id'=>\$session_id |
| Website ID | String | 8-character alpha-numeric | 'website_id'=>\$website_id |
| IP address | String | 16-character alpha-numeric | kount_inquiry 'ip_address'=>\$ip_address |
| Amount | String | 9-character decimal; Up to 7-character numeric + 2-character numeric after the decimal point <div>EXAMPLE: 1234567.89</div> | 'amount'=>\$amount |
| Email | String | 64-character alpha-numeric | 'email'=>\$email |
| Product | Array | Itemnumber + 5 variables required per Product item, as follows Maximum 25 product items | kount_inquiry |
| Shopping cart item type ⁿ | String | 255-character alpha-numeric | n/a |
| Shopping cart item SKUn | String | 255-character alpha-numeric | n/a |

| Value | Type | Limits | Set Method |
|---|--------|---|------------|
| Shopping cart item description ⁿ | String | 255-character alpha-numeric | n/a |
| Shopping cart item quantity ⁿ | String | 5-character numeric | n/a |
| Shopping cart item price ⁿ | String | 9-character decimal; Up to 7-character numeric + 2-character numeric after the decimal point <div>EXAMPLE: 1234567.89</div> | n/a |

Table 2: Kount Inquiry transaction object optional values

| Value | Type | Limits | Set Method |
|--|--------|-----------------------------|------------------------------------|
| Automatic Number Identification (ANI) <div>NOTE: Conditional – must only be sent when Call Center indicator = Y</div> | String | 32-character alpha-numeric | 'auto_number_id'=>\$auto_number_id |
| AVS response | String | 1-character alpha-numeric | 'avs_response'=>\$avs_response |
| CVD response | String | 2-character alpha-numeric | 'cvd_response'=>\$cvd_response |
| Billing street 1 | String | 255-character alpha-numeric | 'bill_street_1'=>\$bill_street_1 |
| Billing street 2 | String | 255-character alpha-numeric | 'bill_street_2'=>\$bill_street_2 |
| Billing country | String | 2-character alpha-numeric | 'bill_country'=>\$bill_country |
| Billing city | String | 255-character alpha-numeric | 'bill_city'=>\$bill_city |

| Value | Type | Limits | Set Method |
|--------------------------|--------|---|---|
| Billing state/province | String | 255-character alpha-numeric | 'bill_province'=>\$bill_province |
| Billing postal/ZIP code | String | 20-character alpha-numeric | 'bill_postal_code'=>\$bill_postal_code |
| Date of birth | String | YYYY-MM-DD | 'dob'=>\$dob |
| Timestamp | String | 10-character alpha-numeric | kount_inquiry 'epoc'=>\$epoc |
| Gender | String | 1-character alpha-numeric M or F | 'gender'=>\$gender |
| Last 4 digits of CC | String | 4-character alpha-numeric | 'last4'=>\$last4 |
| Customer name | String | 64-character alpha-numeric | 'customer_name'=>\$customer_name |
| Financial order ID | String | 64-character alpha-numeric | 'financial_order_id'=>\$financial_order_id |
| Shipping street 1 | String | 255-character alpha-numeric | kount_inquiry 'ship_street_1'=>\$ship_street_1 |
| Shipping street 2 | String | 256-character alpha-numeric | 'ship_street_2'=>\$ship_street_2 |
| Shipping city | String | 255-character alpha-numeric | 'ship_city'=>\$ship_city |
| Shipping email | String | 64-character alpha-numeric | 'ship_email'=>\$ship_email |
| Shipping name | String | 64-character alpha-numeric | 'ship_name'=>\$ship_name |
| Shipping postal/ZIP code | String | 20-character alpha-numeric | 'ship_postal_code'=>\$ship_postal_code |
| Shipping province/state | String | 255-character alpha-numeric | 'ship_province'=>\$ship_province |

| Value | Type | Limits | Set Method |
|--|--------|-----------------------------|------------------------------------|
| Shipping type | String | 2-character alpha-numeric | 'ship_type'=>\$ship_type |
| Customer ID | String | 32-character alpha-numeric | 'customer_id'=>\$customer_id |
| Local attribute <i>n</i> (up to 25 attributes allowed) | String | 255-character alpha-numeric | 'local_attrib_n'=>\$local_attrib_n |
| Data key | String | 25-character alpha-numeric | 'data_key'=>\$data_key |

| Sample Kount Inquiry |
|--|
| <pre> <?php require "../kountClasses.php"; /***** Request Variables *****/ \$store_id=\$argv[1]; \$api_token=\$argv[2]; /***** Transactional Variables *****/ \$type='kount_inquiry'; \$kount_merchant_id=\$argv[3]; \$kount_api_key=\$argv[4]; \$order_id=\$argv[5]; \$call_center_ind=\$argv[6]; \$currency=\$argv[7]; \$email=\$argv[8]; \$auto_number_id=\$argv[9]; \$merchant_ack=\$argv[10]; \$payment_token=\$argv[11]; \$payment_type=\$argv[12]; \$ip_address=\$argv[13]; \$session_id=\$argv[14]; \$website_id=\$argv[15]; \$amount=\$argv[16]; //all the optional fields: 'data_key','customer_id', 'customer_user_agent', 'financial_order_ id','payment_response', //'avs_response','cvd_response','bill_street_1','bill_street_2','bill_country','bill_city','bill_ postal_code','bill_phone', //'bill_province','dob','epoc','gender','last4','customer_name','ship_street_1', 'ship_street_2', 'ship_country', 'ship_city', 'ship_email', //'ship_name', 'ship_postal_code', 'ship_phone', 'ship_province', 'ship_type','prod_type_n','prod_ item_n','prod_desc_n','prod_quant_n','prod_price_n', 'local_attrib_n' //let's put 2 \$avs_response=\$argv[17]; \$cvd_response=\$argv[18]; \$prod_type_1=\$argv[19]; \$prod_item_1=\$argv[20]; \$prod_desc_1=\$argv[21]; \$prod_quant_1=\$argv[22]; \$prod_price_1=\$argv[23]; /***** Transactional Associative Array *****/ </pre> |

Sample Kount Inquiry

```
$txnArray=array (
    'type'=>$type,
    'kount_merchant_id'=>$kount_merchant_id,
    'kount_api_key'=>$kount_api_key,
    'order_id'=>$order_id,
    'call_center_ind'=>$call_center_ind,
    'currency'=>$currency,
    'email'=>$email,
    'auto_number_id'=>$auto_number_id,
    'merchant_ack'=>$merchant_ack,
    'payment_token'=>$payment_token,
    'payment_type'=>$payment_type,
    'ip_address'=>$ip_address,
    'session_id'=>$session_id,
    'website_id'=>$website_id,
    'amount'=>$amount,
    'avs_response'=>$avs_response,
    'cvd_response'=>$cvd_response,
    'prod_type_1'=>$prod_type_1,
    'prod_item_1'=>$prod_item_1,
    'prod_desc_1'=>$prod_desc_1,
    'prod_quant_1'=>$prod_quant_1,
    'prod_price_1'=>$prod_price_1
);
/***** Transaction Object *****/
$kountTxn = new kountTransaction($txnArray);
/***** Request Object *****/
$kountRequest = new kountRequest($kountTxn);
/***** HTTPS Post Object *****/
$kountHttpPost =new kountHttpPost($store_id,$api_token,$kountRequest);
/***** Response *****/
$kountResponse=$kountHttpPost->getkountResponse();
print("\nResponseCode = " . $kountResponse->getResponseCode());
print("\nReceiptId = " . $kountResponse->getReceiptId());
print("\nMessage = " . $kountResponse->getMessage());
print("\nKountResult = " . $kountResponse->getKountResult());
print("\nKountScore = " . $kountResponse->getKountScore());
$kountInfo = $kountResponse->getKountInfo();
foreach($kountInfo as $key => $value)
{
    print("\n".$key ." = ".$value);
}
?>
```

3.3.2 Kount Update

Kount Update transaction object definition

```
$txnArray = array('type'=>'kount_update', ...);
```

```
$mpgTxn = new mpgTransaction($txnArray);
```

HttpPostRequest object for Kount Update transaction

```
$mpgRequest = new mpgRequest($mpgTxn);
```

```
$mpgHttpPost = new mpgHttpPost($store_id,$api_token,$mpgRequest);
```

Kount Update transaction object request values

Table 1: Kount Update transaction object mandatory values

| Value | Type | Limits | Set Method |
|----------------------|--------|----------------------------|--|
| Kount merchant ID | String | 6-character alpha-numeric | 'kount_merchant_id'=>\$kount_merchant_id |
| Kount API key | String | Varies | 'kount_api_key'=>kount_api_key |
| Session ID | String | 32-character alpha-numeric | 'session_id'=>\$session_id |
| Kount transaction ID | String | 32-character alpha-numeric | 'kount_transaction_id'=>\$kount_transaction_id |

Table 2: Kount Update transaction object optional values

| Value | Type | Limits | Set Method |
|---------------------|--------|-------------------------------------|--|
| Payment response | String | 1-character alpha-numeric | 'payment_response'=>\$payment_response |
| AVS response | String | 1-character alpha-numeric | 'avs_response'=>\$avs_response |
| CVD response | String | 2-character alpha-numeric | 'cvd_response'=>\$cvd_response |
| Last 4 digits of CC | String | 4-character alpha-numeric | 'last4'=>\$last4 |
| Financial order ID | String | 64-character alpha-numeric | 'financial_order_id'=>\$financial_order_id |
| Payment token | String | 32-character alpha-numeric | 'payment_token'=>\$payment_token |
| Payment type | String | 4-character alpha-numeric | 'payment_type'=>\$payment_type |
| Refund status | String | 1-character alpha-numeric R or C | 'refund_status'=>\$refund_status |
| Data key | String | 25-character alpha-numeric | 'data_key'=>\$data_key |

Sample Kount Update

```
<?php
require "../kountClasses.php";
/***** Request Variables *****/
$store_id=$argv[1];
$api_token=$argv[2];
/***** Transactional Variables *****/
$type='kount_update';
$kount_merchant_id=$argv[3];
$kount_api_key=$argv[4];
$order_id=$argv[5];
$session_id=$argv[6];
$kount_transaction_id=$argv[7];
$merchant_ack=$argv[8];
//all the optional fields: 'evaluate', 'refund_status', 'avs_response', 'cvd_response', 'payment_
    response', 'last4', 'order_number', 'payment_token', 'payment_type'
//let's put 2
$avs_response=$argv[9];
$cvd_response=$argv[10];
/***** Transactional Associative Array *****/
$txnArray=array (
    'type'=>$type,
    'kount_merchant_id'=>$kount_merchant_id,
    'kount_api_key'=>$kount_api_key,
    'order_id'=>$order_id,
    'session_id'=>$session_id,
    'kount_transaction_id'=>$kount_transaction_id,
    'merchant_ack'=>$merchant_ack,
    'avs_response'=>$avs_response,
    'cvd_response'=>$cvd_response
);
/***** Transaction Object *****/
$kountTxn = new kountTransaction($txnArray);
/***** Request Object *****/
$kountRequest = new kountRequest($kountTxn);
/***** HTTPS Post Object *****/
$kountHttpPost =new kountHttpPost($store_id,$api_token,$kountRequest);
/***** Response *****/
$kountResponse=$kountHttpPost->getkountResponse();
print("\nResponseCode = " . $kountResponse->getResponseCode());
print("\nReceiptId = " . $kountResponse->getReceiptId());
print("\nMessage = " . $kountResponse->getMessage());
print("\nKountResult = " . $kountResponse->getKountResult());
print("\nKountScore = " . $kountResponse->getKountScore());
$kountInfo = $kountResponse->getKountInfo();
foreach($kountInfo as $key => $value)
{
    print("\n".$key ." = ". $value);
}
?>
```

3.4 Kount Risk Inquiry Responses

The risk inquiry response from Kount will have two components, also known as "indications":

- A Kount Risk Score
- An Auto Decision Response Code

3.4.1 The Kount Risk Score

Using the information collected Data Collector and Risk Inquiry Service processes, Kount employs its algorithm to analyze the total risk of a transaction and returns a risk score between 0 and 99 as a response.

In the Moneris Gateway API, the Kount Risk Score is represented by the response field **KountScore**.

3.4.2 Auto Decision Response Codes

Auto decision response codes are a letter code that you receive in the Kount risk inquiry response that are used to automate decision-making after receiving the assessment of risk. You set up the criteria for returning the letter code (and its subsequent automated decision) in Kount's Agent Web Console portal.

The risk inquiry response will contain one of the three letter codes:

A = Accept the transaction

D = Decline the transaction

R = Review the transaction before accepting or declining it

In the Moneris Gateway API, the Auto Decision Response Code is represented by the response field **KountResult**.

4 Testing Your Moneris Kount Services Solution

A testing environment is available for you to connect to while you are integrating your site to the Moneris Gateway. The Moneris Gateway QA environment is generally available 24/7, however since it is a test environment we cannot guarantee 100% availability

When using the APIs in the test environment you will need to use test versions of the Kount Merchant ID, Store ID and API token. The test values of these variables are different than their respective production values.

To begin testing your Moneris Kount Services Solution:

1. Insert the test Kount Merchant ID (obtained from Kount) as the value for **kount_merchant_id**.
2. Insert your test Store ID into the value for **store_id**; for information on how to get your Store ID and API token, see 1 Getting a Unique Test Store ID and API Token
3. Insert your test API token into the value for **api_token**.
4. Change the host URL to the Moneris Gateway test URL at <https://esqa.moneris.com>.

NOTE:

Please be aware that other merchants are using the test environment so you may see transactions and user IDs that you did not create.

As a courtesy to others that are testing we ask that when you are processing refunds, changing passwords and/or trying other functions that you use only the transactions/users that you created.

4.1 Getting Unique Test Store ID and API Token

Transactions requests via the API will require you to have a Store ID and a corresponding API token.

To get your unique Store ID and API token:

1. Log in to the Developer Portal at <https://developer.moneris.com>.
2. In the My Profile dialog, click the Full Profile button.
3. Under My Testing Credentials, select Request Testing Credentials.
4. Enter your Developer Portal password and select your country.
5. Record the Store ID and API token that are given, as you will need them for logging in to the Merchant Resource Center (Store ID) and for API requests (API token).

5 Moving to Production

5.1 Activating a Store for Production

The steps below outline how to activate your production account so that you can process production transactions.

1. Obtain your activation letter/fax from Moneris.
2. Go to <https://www.moneris.com/activate> as instructed in the letter/fax.
3. Input your store ID and merchant ID from the letter/fax and click **Activate**.
4. Follow the on-screen instructions to create an administrator account. This account will grant you access to the Merchant Resource Center.
5. Log into the Merchant Resource Center at <https://www3.moneris.com/mpg> using the user credentials created in step 5.1.
6. Proceed to **ADMIN** and then **STORE SETTINGS**.
7. Locate the API token at the top of the page. Use this API Token along with the store ID that you received in your letter/fax and to send any production transactions through the API.

For more information about how to use the Merchant Resource Center, see the Moneris Gateway Merchant Resource Center User's Guide, which is available at <https://developer.moneris.com>.

5.2 Configuring a Store for Production

After you have completed your testing and have activated your production store, you are ready to point your store to the production host. This requires changing the test values of the Store ID, API token and Kount Merchant ID to their respective production values.

To configure a store for production:

1. Change the value in **kount_merchant_id** to the production Kount Merchant ID.
2. Change the value in **store_id** to the production Store ID that you received when you activated your production store. To review the steps for activating a production store, see *Activating a Production Store Account* (page 1).
3. Change the value in **api_token** to the production API token that you received during activation.
4. Change the host URL to the Moneris Gateway production server at <https://www3.moneris.com>.

6 Getting Help

Moneris has help for you at every stage of the integration process.

| Getting Started | During Development | Production |
|---|--|---|
| <p>Contact our Client Integration Specialists:</p> <p>clientintegrations@moneris.com</p> <p>Hours: Monday – Friday, 8:30am to 8 pm ET</p> | <p>If you are already working with an integration specialist and need technical development assistance, contact our eProducts Technical Consultants:</p> <p>1-866-319-7450</p> <p>eproducts@moneris.com</p> <p>Hours: 8am to 8pm EST</p> | <p>If your application is already live and you need production support, contact Moneris Customer Service:</p> <p>onlinepayments@moneris.com</p> <p>1-888-471-9511</p> <p>Available 24/7</p> |

For additional support resources, you can also make use of our community forums at

<http://community.moneris.com/product-forums/>

Appendix A Definition of Request Fields

Table 1: Definitions of Request Fields - Kount Risk Inquiries API

| Variable Name | Size/Type | Description |
|-------------------|--------------------------------|---|
| kount_merchant_id | 6-character alphanumeric | Merchant ID assigned to merchant by Kount |
| kount_api_key | Varies | API key assigned to merchant by Kount |
| order_id | 50-character alphanumeric | Unique Moneris Gateway transaction number |
| call_center_ind | 1-character alphanumeric (Y/N) | <p>Flag indicating whether risk inquiry request originates from a call center environment</p> <p>If the customer service agents navigate to a separate order entry page that does not collect iframe data: When sending the kount_inquiry request, set call_center_ind to Y</p> <p>If the customer service agents navigate to the same order entry page as the customer: When sending the kount_inquiry request, set call_center_ind to N</p> <p>If RIS call does not originate from a call center environment, set call_center_ind to N</p> |
| currency | 3-character alphanumeric | Country of currency submitted on order |
| email | 64-character alphanumeric | <p>This is the email address submitted by the customer</p> <div> <p>NOTE: If a call center is accepting orders on behalf of customers and the customer does not provide an email address OR the customer does not have an email address, noe-mail@kount.com must be submitted</p> </div> |

| Variable Name | Size/Type | Description |
|--------------------|---------------------------|---|
| customer_id | 32-character alphanumeric | Merchant-assigned account number for the customer |
| auto_number_id | 32-character alphanumeric | <p>Automatic Number Identification (ANI) submitted with order</p> <p>If the ANI cannot be determined, merchant must pass 0123456789 as the ANID</p> <div> <p>NOTE: This field is only valid for phone-to-web requests where customer service agents navigate to a separate order entry page that does not collect iframe data RIS submissions</p> </div> |
| financial_order_id | 64-character alphanumeric | This is a local identifier used by the merchant to identify the transaction, e.g., a purchase order number |
| payment_response | 1-character alphanumeric | <p>Authorization Status returned to merchant from processor</p> <p>A = Authorized</p> <p>D = Declined</p> <p>In orders where payment_response = A will aggregate towards order velocity of the persona while orders where payment_response = D will decrement the velocity of the persona</p> |
| payment_token | 32-character alphanumeric | <p>Payment token submitted by merchant for order (credit card, payer ID, routing/transit, MICR, and account number)</p> <p>If payment_type is set to None then the payment_token value should be left empty (NULL)</p> <p>If the credit card information is not available and a Moneris Vault token is used to process payment set payment_type = CARD and</p> |

| Variable Name | Size/Type | Description |
|---------------|--------------------------|---|
| | | send the token (data key) in the payment_token field. |
| payment_type | 4-character alphanumeric | <p>Payment Type submitted by merchant:</p> <p>APAY - Apple Pay</p> <p>CARD - Credit Card</p> <p>PYPL - PayPal</p> <p>CHEK - Check</p> <p>NONE - None</p> <p>GDMP - Green Dot Money Pack</p> <p>GOOG - Google Checkout</p> <p>BLML - Bill Me Later</p> <p>GIFT - Gift Card</p> <p>BPAY - BPAY</p> <p>NETELLER - Neteller</p> <p>GIROPAY - GiroPay</p> <p>ELV - ELV</p> <p>MERCADE_PAGO - Mercade Pago</p> <p>SEPA - Single Euro Payments Area</p> <p>INTERAC - Interac</p> <p>CARTE_BLEUE - Carte Bleue</p> <p>POLI - POLi</p> <p>Skrill/Moneybookers - SKRILL</p> <p>SOFORT - Sofort</p> <p>If the credit card information is not available and a Moneris Vault token is used to process payment set payment_type = CARD and send the token (data key) in the payment_token field</p> |

| Variable Name | Size/Type | Description |
|----------------------|--|--|
| session_id | 32-character alphanumeric | Unique Session ID; must be unique over a 30-day span |
| website_id | 8-character alphanumeric | Website identifier of where order originated |
| ip_address | 16-character alphanumeric | <p>Dotted Decimal IPv4 address that the merchant sees coming from the customer</p> <p>If the Phone to Web exclusion is used, the ip_address field should be hard coded to be 10.0.0.1.</p> <p>Other than for Phone to Web requests (where the customer service agents navigate to a separate order entry page that does not collect iframe data), the ip_address field should never be an anonymous IP address (e.g., 10.X.X.X or 192.168.X.X)</p> |
| amount | 9-character decimal; Up to 7-character numeric + 2-character numeric after the decimal point <div>EXAMPLE: 1234567.89</div> | <p>Amount of the transaction</p> <p>Must contain 3 digits with two penny values</p> <p>The minimum value passed can be 0.01 and the maximum 9999999.99</p> |
| kount_transaction_id | 32-character alphanumeric | Transaction ID required for update calls to Kount; generated by Kount, must match KountTransactionId returned from original RIS Post |
| evaluate | 5-character alphanumeric | <p>If set to 'true', full re-evaluation will be performed with Kount</p> <p>If unset, default value is 'false'</p> |
| avs_response | 1-character alphanumeric | Address Verification System verification response returned to merchant from processor with payment request |
| cvd_response | 2-character alphanumeric | CVD verification response returned to merchant from processor with |

| Variable Name | Size/Type | Description |
|------------------|----------------------------|---|
| | | payment request |
| bill_street_1 | 255-character alphanumeric | Billing street address - Line 1 |
| bill_street_2 | 255-character alphanumeric | Billing street address - Line 2 |
| bill_country | 2-character alphanumeric | Billing address - Country |
| bill_city | 255-character alphanumeric | Billing address - City |
| bill_postal_code | 20-character alphanumeric | Billing address - Postal Code |
| bill_phone | 32-character alphanumeric | Bill-to Phone Number |
| bill_province | 255-character alphanumeric | Billing address - State/Province |
| dob | YYYY-MM-DD | Date of Birth |
| epoc | 10-character alphanumeric | This is a timestamp and is expressed as a number; represents the number of seconds elapsed since midnight 01/01/1970 For more information, refer to www.epochconverter.com |
| gender | 1-character alphanumeric | M or F |
| last4 | 4-character numeric | Last 4 digit of the customer's credit card number |
| customer_name | 64-character alphanumeric | Name submitted with the order |
| refund_status | 1-character alphanumeric | Refund/Chargeback status R = Refund C = Chargeback |
| ship_street_1 | 255-character alphanumeric | Shipping street address - Line 1 |
| ship_street_2 | 256-character alphanumeric | Shipping street address - Line 2 |
| ship_country | 2-character alphanumeric | Shipping address - Country |
| ship_city | 255-character alphanumeric | Shipping address - City |

| Variable Name | Size/Type | Description |
|------------------|----------------------------|---|
| ship_email | 64-character alphanumeric | Shipping address - Email address of recipient |
| ship_name | 64-character alphanumeric | Shipping address - Name of recipient |
| ship_postal_code | 20-character alphanumeric | Shipping address - Postal Code |
| ship_phone | 32-character alphanumeric | Ship-to Phone Number |
| ship_province | 255-character alphanumeric | Shipping address - State/Province |
| ship_type | 2-character alphanumeric | <p>Shipping type</p> <p>The following nomenclature is expected for shipping types to be passed to Kount:</p> <p>Same Day = SD</p> <p>Next Day = ND</p> <p>Second Day = 2D</p> <p>Standard = ST</p> <p>NOTE: These three attributes can be used to pass custom attribute data if you want to correlate some data with the returned response</p> |
| prod_type_n | 255-character alphanumeric | <p>Shopping cart data array attribute high level or generalized description of the item added to the shopping cart;</p> <p>NOTE: This value should be passed as plain text, free from any markup or Unicode values</p> |
| prod_item_n | 255-character alphanumeric | <p>Shopping cart data array attribute typically the SKU for an item</p> <p>NOTE: This value should be passed as plain text, free from any markup or Unicode values</p> |

| Variable Name | Size/Type | Description |
|----------------|--|---|
| prod_desc_n | 255-character alphanumeric | Shopping cart data array attribute for a specific description of the item being purchased |
| prod_quant_n | 5-character numeric | Shopping cart data array attribute signifying the quantity of the item being purchased |
| prod_price_n | 9-character decimal; Up to 7-character numeric + 2-character numeric after the decimal point <div>EXAMPLE: 1234567.89</div> | Shopping cart data array attribute for the price of the single item Must contain 3 digits with two penny values The minimum value passed can be 0.01 and the maximum 9999999.99 |
| local_attrib_n | 255-character alphanumeric | Up to 25 of these can be submitted in one request, where n defines each user defined field. Value of n can be 1 to 25 |

Appendix B Definitions of Response Fields

Table 1: Definitions of Response Fields - Kount Risk Inquiries API

| Variable Name | Size/Type | Description |
|--------------------|----------------------------|--|
| ResponseCode | 3-character numeric | Three digit number; positive response when it is less than 50 |
| ReceiptId | 64-character alphanumeric | Echoed order_id in the initial request |
| Message | 255-character alphanumeric | Brief description message for this query |
| KountResult | 1-character alphanumeric | Auto-decision response code: A = Approve D = Decline R = Review |
| KountScore | 3-character numeric | Final risk score returned from Kount system |
| KountTransactionId | 12-character alphanumeric | Kount Transaction ID |
| Browser | 64-character alphanumeric | Web Browser |
| CardBrand | 4-character alphanumeric | Brand of credit card used when payment type is 'credit card' |
| NumCards | numeric | Total number of credit cards associated to persona as seen by Kount |
| Cookies | Y/N | A flag to indicate if device placing order has cookies enabled or not |
| CountersTriggered | numeric | Number of unique counter names triggered during rules evaluation |

| Variable Name | Size/Type | Description |
|---------------|--|--|
| CounterNameX | 64-character alphanumeric | Name of counter triggered |
| CounterValueX | numeric | Sum of the number of times a counter was triggered |
| Country | 2-character alphanumeric | Two-character ISO country code associated with the physical device |
| DDFS | 10-character alphanumeric (YYYY-MM-DD format) | Date device first seen |
| DeviceLayers | 55-character alphanumeric | 5 device layers that comprise the device's fingerprint, representing OS, browser, Javascript, cookies and Flash settings |
| Devices | numeric | Total number of unique devices associated to persona as seen by Kount |
| DSR | 10-character alphanumeric | Device Screen Resolution |
| Emails | numeric | Total number of unique email addresses associated to persona as seen by Kount |
| ErrorCount | numeric | Number of errors the Kount request generated |
| ErrorCode | 4-character alphanumeric | Error Code displayed in RIS response |
| Fingerprint | 32-character alphanumeric | The unique fingerprint of the device placing the order |
| Flash | 1-character alphanumeric (Y/N) | A flag to indicate if the device placing the order has Flash enabled |
| Geox | 2-character alphanumeric | Persona related country with highest probability of fraud |
| HttpCountry | 2-character alphanumeric | User home country the device owner has set in the device's control panel |

| Variable Name | Size/Type | Description |
|-----------------|-----------------------------------|---|
| ProxyAddress | 16-character alphanumeric | IP address of proxy |
| JavaScript | 1-character alphanumeric (Y/N) | A flag to indicate if the device placing order has JavaScript enabled |
| DataCollector | 1-character alphanumeric (Y/N) | Flag indicating whether or not device data was collected by the Data Collector process |
| KYCF | n/a | Know Your Customer Flag |
| Language | 2-character alphanumeric | The language the device owner has set in the device control panel |
| LocalTime | 20-character alphanumeric | The local time the device owner has set in the device control panel |
| KountMerchantID | numeric | Kount Merchant ID |
| MobileDevice | 1-character alphanumeric (Y/N) | Flag to indicate if device placing the order is of mobile nature |
| MobileForwarder | 1-character alphanumeric (Y/N) | If device is mobile, flag that indicates whether it is using a forwarder to process the carrier's service |
| MobileType | 32-character alphanumeric | Type of mobile device, e.g., iPhone, Android, Blackberry, iPad, etc. |
| Network | 1-character alphanumeric | Riskiest network type associated with persona within last 14 days: A = Anonymous H = High School L = Library N = Normal O = Open Proxy |

| Variable Name | Size/Type | Description |
|----------------|-----------------------------------|--|
| | | P = Prison S = Satellite |
| Mode | 1-character alphanumeric | Specifies the RIS post mode type |
| OS | 64-character alphanumeric | Operating System of the device |
| PCRemote | 1-character alphanumeric (Y/N) | Flag indicating whether the device is enabled to use PC remote software |
| Proxy | 1-character alphanumeric | Flag indicating whether or not a proxy server is detected |
| PiercedAddress | 16-character alphanumeric | Pierced IP address PIP_COUNTRY = Country of pierced IP address (2, US) PIP_LAT = Latitude of pierced IP address (Number, -90.1922) PIP_LON = Longitude of pierced IP address (Number, 38.6312) PIP_CITY = City of pierced IP address (255, Houston) PIP_REGION = State/Region of pierced IP address (255 character limit) PIP_ORG = Owner of pierced IP address or address block (64, Organization Name) |
| ReasonCode | 16-character alphanumeric | Reason code associated with rule action |
| Region | 2-character alphanumeric | Region associated to device location |
| GeoxRegion | 2-character alphanumeric | Region associated to GEOX Location |
| RulesTriggered | numeric | Number of rules triggered by post to Kount |

| Variable Name | Size/Type | Description |
|------------------|--------------------------------|---|
| RuleIdX | numeric | Rule ID associated with merchant-created rules |
| RuleDescriptionX | 255-character alphanumeric | Rule descriptions associated with RuleIdX |
| SessionId | 32-character alphanumeric | Unique Session ID |
| Website | 8-character alphanumeric | Website identifier of where order originated |
| Timezone | 6-character numeric | The time zone of the device; value is number of minutes divided by 60 from Greenwich Mean Time |
| UserAgent | 1024-character alphanumeric | User agent string |
| Velocity | numeric | Quantity of orders seen from persona within last 14 days |
| MaxVelocity | numeric | Quantity of orders from persona within the most active 6 hour window in last 14 days; payment_response field must be equal to 'A' |
| Version | 4-character alphanumeric | Specifies version of Kount system being used |
| VoiceDevice | 1-character alphanumeric (Y/N) | If it is a mobile device, flag indicating whether the device is voice activated |
| WarningCount | numeric | Number of warnings the RIS post generated |
| WarningCode | 4-character alphanumeric | Warning code displayed in the RIS response |

Appendix C Error Messages

Error messages that are returned if the gateway is unreachable

Global Error Receipt

You are not connecting to our servers. This can be caused by a firewall or your internet connection.

Response Code = NULL

The response code can be returned as null for a variety of reasons. The majority of the time, the explanation is contained within the Message field.

When a 'NULL' response is returned, it can indicate that the issuer, the credit card host, or the gateway is unavailable. This may be because they are offline or because you are unable to connect to the internet.

A 'NULL' can also be returned when a transaction message is improperly formatted.

Error messages that are returned in the Message field of the response

XML Parse Error in Request: <System specific detail>

An improper XML document was sent from the API to the servlet.

XML Parse Error in Response: <System specific detail>

An improper XML document was sent back from the servlet.

Transaction Not Completed Timed Out

Transaction timed out before the host responds to the gateway.

Request was not allowed at this time

The host is disconnected.

Could not establish connection with the gateway: <System specific detail>

Gateway is not accepting transactions or server does not have proper access to internet.

Input/Output Error: <System specific detail>

Servlet is not running.

The transaction was not sent to the host because of a duplicate order id

Tried to use an order id which was already in use.

The transaction was not sent to the host because of a duplicate order id

Expiry Date was sent in the wrong format.

Vault error messages

Can not find previous

Data key provided was not found in our records or profile is no longer active.

Invalid Transaction

Transaction cannot be performed because improper data was sent.

or

Mandatory field is missing or an invalid SEC code was sent.

Malformed XML

Parse error.

Incomplete

Timed out.

or

Cannot find expiring cards.

Custom Android Pay In-App SDK Responses

| Response Code | Message | Definition |
|---------------|--------------|---------------------------|
| 900 | Global Error | Unable to decrypt payload |

Appendix D Response Codes for Kount

| Response Code | Response Message |
|---------------|---|
| 001 | Success |
| 973 | Unable to locate merchant kount details |
| 984 | Data error |
| 987 | Invalid transaction |

For Error Codes specific to Kount, see the Kount portal at

https://support.kount.com/Developer_Resources/Getting_Started/Technical_Specification_Guide_r._January_2016_-_v_6.4.5#Appendix_C

Appendix E Security Requirements

All Merchants and Service Providers that store, process, or transmit cardholder data must comply with PCI DSS and the Card Association Compliance Programs. However, validation requirements vary by business and are contingent upon your "Merchant Level" or "Service Provider Level". Failure to comply with PCI DSS and the Card Association Compliance Programs 3.2 may result in a Merchant being subject to fines, fees or assessments and/or termination of processing services. Non-compliant solutions may prevent merchants boarding with Moneris Solutions.

As a Moneris Solutions client or partner using this method of integration, your solution must demonstrate compliance to the Payment Card Industry Data Security Standard (PCI DSS) and/or the Payment Application Data Security Standard (PA DSS) 3.2. These standards are designed to help the cardholders and merchants in such ways as they ensure credit card numbers are encrypted when transmitted/stored in a database and that merchants have strong access control measures, logging, secure software updates, secure remote access and support.

For further information on PCI DSS and PA DSS requirements, please visit www.pcisecuritystandards.org.

For more information on how to get your application PCI-DSS compliant, please contact our Integration Specialists and visit <https://developer.moneris.com> to view the PCI-DSS Implementation Guide.