

Payment-engine.com |

# Payment-Engine

## API/SDK Documentation

PAYMENT ENGINE

**Document Version 2.07.0415**

Copyright © 2001-07 Blue Bamboo. All Rights Reserved

# Introduction

Payment-engine.com |

Visit us at: <https://secure.payment-engine.com>.

Copyright © 2001-07 Payment-Engine

Congratulations on the selection of the PAYMENT ENGINE Software Development Kit (SDK) Payment Processing Software, the most advanced solution in the industry for processing credit cards, debit cards and check services. This software provides you with a fast, easy, reliable way to authorize credit card, ATM/debit card, and check transactions on your PC. This guide prepares you with the detailed information that you will need to install, configure and test your payment processing application.

Your opinion is important to us. If you have any suggestions feel free to [email us](#).

**Thank you for choosing the PAYMENT ENGINE!**

## Support

Blue Bamboo is committed to providing the highest quality tools and customer support. If you have any questions, comments or suggestions please contact Blue Bamboo by:

Online: <http://support.payment-engine.com>

## **License Information**

### ***DISCLAIMER***

The SmartPayments and supporting services are licensed on an “as is” basis. There are no warranties, expressed or implied, including, but not limited to, warranties of merchantability or of fitness for a particular purpose, and all such warranties are expressly and specifically disclaimed. Blue Bamboo shall have no liability or responsibility to you nor any other person or entity with respect to any liability, loss, or damage caused or alleged to be caused directly or indirectly by any SmartPayments software and supporting services. Use of the SmartPayments software and supporting services signifies agreement with this disclaimer and is subject to the license agreement provided with the installation of the software.

### **LICENSE**

Use of the SmartPayments services requires a signed service agreement by the end user of the service or reseller of such services. Blue Bamboo grants you the right to use this software and supporting software for the purpose of connecting to the BLUE BAMBOO payment gateway or your processor, on any device of choice. You may physically transfer each license, without cost, to a different computer.

### **RESTRICTED USE**

Copying of the manual, interface specifications, or system files for use other than backing up files and/or to connect to systems other than the BLUE BAMBOO payment gateway or your processor is prohibited. You may not remove any product identification, copyright, or other proprietary notices from the software or documentation. Reverse engineering is strictly prohibited.

This client software is provided for use in connecting with the BLUE BAMBOO payment system gateway or your processor only. Use of any components, code, or documentation to connect to other systems is strictly prohibited and a violation of this agreement and subject to all remedies available by law.

### ***EULA***

#### **END-USER LICENSE AGREEMENT FOR SMARTPAYMENTS SOFTWARE**

##### **IMPORTANT READ CAREFULLY:**

This Blue Bamboo End-User License Agreement ( EULA ) is a legal agreement between you (either an individual or a single entity) and Blue Bamboo for the Blue Bamboo product accompanying this EULA, which includes computer software and may include associated media, printed materials, and online or electronic documentation (SOFTWARE). By installing, copying, or otherwise using the SOFTWARE, you agree to

be bound by the terms of this EULA. If you do not agree to the terms of this EULA, do not install, copy, or otherwise use the SOFTWARE.

#### SOFTWARE PRODUCT LICENSE

Blue Bamboo Software (SOFTWARE) is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE is licensed, not sold.

#### GRANT OF LICENSE

Blue Bamboo, hereby grants you ("Licensee") a limited, nonexclusive, non-transferable, royalty-free license to make and use a single copy an unlimited number of copies of the SOFTWARE accompanying this EULA to be installed on CPUs residing on Licensee's premises, solely for Licensee's internal use. Blue Bamboo and its suppliers shall retain title and all ownership rights to the product, and this Agreement shall not be construed in any manner as transferring any rights of ownership or license to the SOFTWARE or to the features or information therein, except as specifically stated herein.

#### USE RESTRICTION

Licensee acknowledges that the SOFTWARE acquired hereunder can only be used for reseller and/or end merchant evaluation and/or product's intended use of payment processing. Use of this product for any other purposes such as Competitor Evaluation, Reverse Engineering, Decompilation, and Disassembly is violation of this License and Licensee agrees that such acts are a blatant and flagrant violation of this License and will be subject to any and all penalties and remedies available by Law for violation of intellectual property laws and treaties.

#### DESCRIPTION OF OTHER RIGHTS AND LIMITATIONS

(a) Limitations on Modification, Reverse Engineering, Decompilation, and Disassembly. Licensee may not modify, reverse engineer, decompile, or disassemble the SOFTWARE, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.

(b) No Rights to Sublicense or Rental. Licensee may not rent, lease or lend the SOFTWARE.

(c) Termination. Without prejudice to any other rights, Blue Bamboo may terminate this EULA if Licensee fails to comply with the terms and conditions of this EULA. In such event, Licensee must destroy all copies of the SOFTWARE and all of its component parts.

(d) Reservation of Rights. Licensee agrees that the SOFTWARE is owned Blue Bamboo and all rights not expressly granted herein are reserved by Blue Bamboo.

## PRODUCT MAINTENANCE

Licensee understands and agrees that Blue Bamboo may provide updates to the SOFTWARE from time to time but Blue Bamboo shall have no obligation to provide maintenance or updates to Licensee for SOFTWARE licensed under this Agreement.

## CONFIDENTIALITY

(a) Licensee understands that the SOFTWARE contains confidential and proprietary trade secret information of Blue Bamboo that are not commercially available to the public. The Licensee agrees that, in partial consideration for Blue Bamboo's allowing Licensee access to and use of the SOFTWARE pursuant to this EULA:

(i) Licensee shall treat the SOFTWARE in the same manner that it treats its most confidential and proprietary trade secret materials, and

(ii) Licensee shall take all measures necessary to prevent the SOFTWARE from falling into the possession of persons not bound to maintain the confidentiality of the trade secrets contained within the SOFTWARE. As such, Licensee shall only permit employees and contractors who have a need to use the SOFTWARE for the purposes stated in the license grant (Section 1) to have access to and use such SOFTWARE, and Licensee's contractors shall not use the SOFTWARE unless and until they have entered into written non-disclosure agreements with the Licensee that require them to maintain the confidentiality of SOFTWARE. Licensee shall promptly advise Blue Bamboo, in writing, of any misappropriation or misuse of the SOFTWARE by any person which may come to the Licensee's attention.

(b) Licensee understands and agrees that disclosure or use of the SOFTWARE except as authorized above will result in irreparable harm to Blue Bamboo and that monetary damages may be inadequate to compensate Blue Bamboo for such breach. Accordingly, the Licensee agrees that Blue Bamboo will, in addition to any other remedies available to it at law or in equity, be entitled to injunctive relief to enforce the terms of this Agreement.

## COPYRIGHT

All title and copyrights in and to the SOFTWARE (including but not limited to any images, photographs, animations, video, audio, music, text, and applets incorporated into the SOFTWARE), the accompanying printed materials, and any copies of the SOFTWARE are owned by Blue Bamboo or its suppliers. The SOFTWARE is protected

by copyright laws and international treaty provisions. Therefore, Licensee must treat the SOFTWARE PRODUCT like any other copyrighted material except that Licensee may install the SOFTWARE on a single computer provided Licensee keep the original solely for backup or archival purposes. Licensee may not copy any printed materials accompanying the SOFTWARE without prior permission of Blue Bamboo.

#### U.S. GOVERNMENT RESTRICTED RIGHTS

The SOFTWARE and documentation are provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1) and (2) of the Commercial Computer Software Restricted Rights at 48 CFR 52.227-19, as applicable. Manufacturer is Blue Bamboo./101 NE 3rd Ave., Suite 1500/Ft. Lauderdale, Fl 33301.

#### EXPORT RESTRICTIONS

Licensee acknowledges that the SOFTWARE acquired hereunder are subject to the export control laws and regulations of the U.S.A., and any amendments thereof. Licensee confirms that with respect to these SOFTWARE, it will not export or re-export them, directly or indirectly, either to (i) any countries that are subject to U.S.A export restrictions (currently including, but not necessarily limited to, Cuba, the Federal Republic of Yugoslavia (Serbia and Montenegro), Iran, Iraq, Libya, North Korea, South Africa (military and police entities), Syria, and Vietnam); (ii) any end user who Licensee knows or has reason to know will utilize them in the design, development or production of nuclear, chemical or biological weapons; or (iii) any end user who has been prohibited from participating in the U.S.A. export transactions by any federal agency of the U.S.A. government. Licensee further acknowledges that the SOFTWARE may include technical data subject to export and re-export restrictions imposed by U.S.A. law.

#### DISCLAIMER OF WARRANTY

THE SOFTWARE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, BLUE BAMBOO FURTHER DISCLAIMS ALL WARRANTIES, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. THE ENTIRE RISK ARISING OUT OF THE USE OR PERFORMANCE OF THE PRODUCT AND DOCUMENTATION REMAINS WITH LICENSEE.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL BLUE BAMBOO BE LIABLE FOR ANY CONSEQUENTIAL, INCIDENTAL, DIRECT, INDIRECT, SPECIAL, PUNITIVE, OR OTHER DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS

INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE PRODUCT OR DOCUMENTATION, EVEN IF BLUE BAMBOO HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME STATES/JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO LICENSEE.

## GOVERNING LAW

This EULA shall be governed by the laws of the State of Washington.

Should you have any questions concerning this EULA, or if you desire to contact Blue Bamboo for any reason, please write: Blue Bamboo./101 NE 3rd Ave., Suite 1500/Ft. Lauderdale, Fl 33301.

## Overview

### *Testing*

You can request a test account on our server to be established at our demo host server. To request this account, please send your email request to our technical support staff. Please include the following information with your test account request: company name, your name, phone number, email address associated with the test account, and which payment processor you would like to test. An email response will be sent with valid test information.

Testing can be performed with the following test cards:

| Card Type  | Number           |
|------------|------------------|
| MasterCard | 5000300020003003 |
| Visa       | 4005550000000019 |
| Discover   | 6001111111111117 |
| Diners     | 36999999999999   |
| AMEX       | 374255312721002  |

## Web Services

### ePayment Web Services

The following operations are supported.

- [GetCardTrx](#) – Retrieves card transaction details for a merchant
- [GetCardTrxSummary](#) – Retrieves card transaction summary for a merchant
- [GetCheckTrx](#) – Retrieves check transaction details for a merchant
- [GetCardType](#) – Returns the name of the card issuer; such as Visa, MasterCard, AMEX, etc.
- [GetInfo](#) – Retrieves information from the web service
- [GetOpenBatchSummary](#) – Retrieves payment type transaction summary of the current open batch for a merchant
- [IsCommercialCard](#) – Returns (T/F) if the card is a known commercial card
- [ProcessCheck](#) – Processes check transactions for a merchant
- [ProcessCreditCard](#) – Processes credit card transactions for a merchant
- [ProcessDebitCard](#) – Processes debit card transactions for a merchant
- [ProcessEBTCard](#) – Processes EBT card transactions for a merchant
- [ProcessGiftCard](#) – Processes gift card transactions for a merchant
- [ProcessSignature](#) – Sends a signature to apply to a receipt transaction
- [ValidCard](#) – Validates the credit card by checking the card length based on the card type, performing a mod 10 checksum, and validating the expiration date
- [ValidCardLength](#) – Validates the credit card length
- [ValidExpDate](#) – Validates the expiration date of the credit card
- [ValidMod10](#) – Validates the credit card by performing a mod 10 checksum on the card number; returns (T/F)
- [AddMerchant](#) – Adds a merchant to the account
- [DeleteMerchant](#) – Deletes a merchant from the account
- [AddRecurringCreditCard](#) -Allows customer information to be programmatically stored through web services for recurring billing.
- [AddRecurringCheck](#)- Allows check information to be programmatically stored through web services for recurring billing.
- [\\*ProcessCreditCard](#) -Allows for processing credit card transactions in recurring billing .
- [\\*ProcessCheck](#)- Allows for processing ACH /check transactions for recurring billing.
- [ManageCheckInfo](#) – Allows for programmatic management of existing check information for recurring billing.

- [ManageCreditCardInfo](#)- Allows for programmatic management of credit card information for customers specific to recurring billing.
- [ManageContract](#) – Allows for managing existing contracts for updates and modifications.
- [ManageCustomer](#) – Allows for management of existing customers in the recurring billing web service.
- [ManageContractAddDaysToNextBillDt](#) – Allows for modification of next billing date for recurring billing contracts.
- [GetNetworkID](#)- This web service allows for returning the debit network ID if the debit card number matches any of these network’s bin ranges

**\* = This is for recurring.asmx webservice. Not to be confused with Transact.asmx**

## GetCardTrx

This Web service operation retrieves card transaction details for a merchant. The URL to access this Web service is: <https://localhost/admin/ws/trxdetail.asmx?op=GetCardTrx>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter       | Description  |
|-----------------|--|
| <b>UserName</b> | Required. User name assigned in the payment server   |
| <b>Password</b> | Required. Password for the user name assigned in the payment server  |
| <b>RPNum</b>    | Required. The merchant's RP Number in order to uniquely identify merchant's account for the query  |
| <b>PNRef</b>    | Optional. The unique payment reference number assigned to the transaction. <i>If this field is provided, all other query fields will be ignored when using <b>PNRef</b> parameter to query the system.</i>   |
| <b>BeginDt</b>  | Required, except when PNRef is provided. The begin date of the date range in MM/DD/YYYY (or YYYY-MM-DD or YYYY-MM-DDThh:mm:ss) format. This date will be converted to YYYY-MM-DDThh:mm:ss (time is in 24-hour format). If the passed-in BeginDt does not contain time information, BeginDt will default to midnight on the given date. For example:<br>2005-08-19T12:00:12 is kept as is<br>2005-08-19 becomes 2005-08-19T00:00:00<br>2005/08/19 becomes 2005-08-19T00:00:00<br>08/19/2005 becomes 2005-08-19T00:00:00<br>The query to obtain transactions in the date range is constructed as follows:<br>(Date DT >=BeginDt) AND (Date DT<EndDt)<br>where Date DT is the transaction timestamp |
| <b>EndDt</b>    | Required, except when <b>PNRef</b> is provided. The end date of the date range in MM/DD/YYYY (or YYYY-MM-DD or YYYY-MM-DDThh:mm:ss) format.  |

|                                  |  |
|----------------------------------|--|
|                                  | <p>This date will be converted to YYYY-MM-DDThh:mm:ss (time is in 24-hour format). If the passed-in EndDt does not contain time information, EndDt will be incremented to the <i>next day</i> at midnight such that no transaction on the desired end date will be excluded based on its time. For example:<br/> 2005-08-19T12:00:12 is kept as is<br/> 2005-08-19 becomes 2005-08-20T00:00:00<br/> 08/19/2005 becomes 2005-08-20T00:00:00</p>   |
| <p><b>PaymentType</b></p>        | <p>Optional. If provided, only those transactions matching the <b>PaymentType</b> will be included. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>'AMEX'</b> American Express card</li> <li>• <b>'CARTBLANCH'</b> Carte Blanch card</li> <li>• <b>'DEBIT'</b> Debit card</li> <li>• <b>'DINERS'</b> Diners Club card</li> <li>• <b>'DISCOVER'</b> Novus Discover card</li> <li>• <b>'EBT'</b> Electronic Benefit Transfer</li> <li>• <b>'JAL'</b> JAL card</li> <li>• <b>'JCB'</b> Japanese Commercial Bank card</li> <li>• <b>'MASTERCARD'</b> Master card</li> <li>• <b>'VISA'</b> Visa card</li> <li>• <b>'EGC'</b> Gift card</li> <li>• <b>'PAYRECEIPT'</b> to retrieve receipt images that were uploaded to the payment server</li> <li>• <b>'SETTLE'</b> to retrieve requests to settle transactions</li> </ul> <p>Or any permutation of the above values, e.g. <b>""PAYRECEIPT', 'SETTLE""</b> will pull all transactions with either PayReceipt or Settle payment types.</p> |
| <p><b>ExcludePaymentType</b></p> | <p>Optional. If provided, any transaction matching the <b>ExcludePaymentType</b> will be excluded</p>  |
| <p><b>TransType</b></p>          | <p>Optional. If provided, only those transactions matching the <b>TransType</b> will be included. Valid values are</p> <ul style="list-style-type: none"> <li>• <b>'Authorization'</b> to retrieve previously-authorized (pre-auth) transactions</li> <li>• <b>'Capture'</b> to retrieve captured transactions</li> <li>• <b>'Credit'</b> to retrieve return transactions</li> <li>• <b>'ForceCapture'</b> to retrieve force-auth transactions</li> <li>• <b>'GetStatus'</b> to make an inquiry to the EBT or gift card's balance</li> <li>• <b>'PostAuth'</b> to retrieve post-auth transactions</li> <li>• <b>'Purged'</b> to remove a transaction from the current batch due to an error</li> <li>• <b>'Receipt'</b> to retrieve receipt images that were uploaded to the payment server</li> <li>• <b>'RepeatSale'</b> to retrieve repeat-sale transactions</li> </ul>   |

|                         |  |
|-------------------------|--|
|                         | <ul style="list-style-type: none"> <li>• <b>'Sale'</b> to retrieve sale transactions</li> <li>• <b>'Void'</b> to retrieve void transactions</li> </ul> <p>Or any permutation of the above values, e.g. <b>""Credit','Sale""</b> will pull all transactions with either Credit or Sale transaction types.</p>   |
| <b>ExcludeTransType</b> | Optional. If provided, any transaction matching the <b>ExcludeTransType</b> will be excluded   |
| <b>ApprovalCode</b>     | Optional. If provided, only those transactions matching the <b>ApprovalCode</b> parameter will be included   |
| <b>Result</b>           | <p>Optional. If provided, only those transactions matching the <b>Result</b> will be included. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>0</b> is Approved</li> <li>• Anything else is declined; if you want all the declined transactions, you should leave this field empty and use the <b>ExcludeResult=0</b> instead.</li> </ul>   |
| <b>ExcludeResult</b>    | Optional. If provided, any transactions matching the <b>ExcludeResult</b> will be excluded.  |
| <b>NameOnCard</b>       | Optional. Cardholder's name as it appears on the card. If provided, only those transactions with cardholder's name matching <b>NameOnCard</b> will be included. Matching is done using wild cards: e.g. "test" will match "test", "1test" and "1test234"   |
| <b>CardNum</b>          | Optional. A card number. If provided, only those transactions with the cardholder's name matching <b>CardNum</b> will be included. Matching is done using wild cards   |
| <b>CardType</b>         | <p>If provided, only those transactions matching the <b>CardType</b> will be included. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>'AMEX'</b> American Express card</li> <li>• <b>'CARTBLANCH'</b> Carte Blanch card</li> <li>• <b>'DEBIT'</b> Debit card</li> <li>• <b>'DINERS'</b> Diners Club card</li> <li>• <b>'DISCOVER'</b> Novus Discover card</li> <li>• <b>'EBT'</b> Electronic Benefit Transfer</li> <li>• <b>'JAL'</b> JAL card</li> <li>• <b>'JCB'</b> Japanese Commercial Bank card</li> <li>• <b>'MASTERCARD'</b> Master card</li> <li>• <b>'VISA'</b> Visa card</li> <li>• <b>'EGC'</b> Gift card</li> </ul> <p>Or any permutation of the above values, <b>""VISA','MASTER','DISCOVER""</b> will pull all transactions with either VISA, MASTER and DISCOVER card type</p> <p>There are cases when Cardtype needs to be set to ALL, for example CardType=ALL</p> |

|                        |  |
|------------------------|--|
| <b>ExcludeCardType</b> | Optional. If provided, any transaction matching the <b>ExcludeCardType</b> will be excluded  |
| <b>ExcludeVoid</b>     | Required, except when <b>PNRef</b> is provided. An option to exclude voided transactions or not; must either be <b>TRUE</b> or <b>FALSE</b>  |
| <b>User</b>            | Optional. The user who originated the transactions. If provided, only those transactions created by the matching <b>User</b> will be included. Matching is done using wild cards   |
| <b>InvoiceId</b>       | Optional. The invoice ID that was included in the original transaction. If provided, only those transactions with matching <b>invoiceId</b> will be included. Matching is done using wild cards  |
| <b>SettleFlag</b>      | Optional. An option to retrieve the settled transactions or unsettled transactions; must either be <b>1</b> for true or <b>0</b> for false   |
| <b>SettleMsg</b>       | Optional. The settlement ID or message returned from the host  |
| <b>SettleDt</b>        | Optional. The date of the settlement   |
| <b>TransformType</b>   | Optional. The type of format to transform the data into. Leave the field blank to default to <b>XML</b> <ul style="list-style-type: none"> <li>• <b>XML</b> will output the plain XML string</li> <li>• <b>XSL</b> will use XSL to transform the XML output</li> <li>• <b>DELIM</b> uses <b>ColDelim</b> and <b>RowDelim</b> to format the output</li> </ul>   |
| <b>Xsl</b>             | Optional. This field is used only if the <b>TransformType</b> is <b>XSL</b> . The XSL to transform the resulting dataset. If provided, the resulting dataset will be transformed using this XSL. You may pass in a URL to the XSL file, or the XSL string itself. If this field is not empty, the Web Services will try to locate the file from the URL. If that also fails, it will treat it as an XSL string. In any case, the final XSL string will be loaded and validated against the XSL schema; if it passes, then that XSL will be used for transformation. A sample predefined XSL is included with this Web Services: <ul style="list-style-type: none"> <li>• <b>http://localhost/admin/ws/TabDelim.xsl</b> for a tab delimited transformation</li> </ul> |
| <b>ColDelim</b>        | Optional. This field is used only if the <b>TransformType</b> is <b>DELIM</b> . This defines the string that separates each column   |
| <b>RowDelim</b>        | Optional. This field is used only if the <b>TransformType</b> is <b>DELIM</b> . This defines the string that separates each row  |
| <b>IncludeHeader</b>   | Optional. This field is used only if the <b>TransformType</b> is <b>DELIM</b> . If <b>TRUE</b> , then field headers will be included in the first row using the same delimiter strings; must either be <b>TRUE</b> or <b>FALSE</b>   |
| <b>ExtData</b>         | Optional. Extended data in XML format. Valid values are: <ul style="list-style-type: none"> <li>• <b>&lt;IMAGE_TYPE&gt;NO_IMAGE&lt;/IMAGE_TYPE&gt;</b> for no image</li> <li>• <b>&lt;IMAGE_TYPE&gt;ONLY_IMAGE&lt;/IMAGE_TYPE&gt;</b> for only the image</li> <li>• <b>&lt;IMAGE_TYPE&gt;ALL&lt;/IMAGE_TYPE&gt;</b> for all images</li> <li>• <b>&lt;CustomerID&gt;CustomerID&lt;/CustomerID&gt;</b> for customer ID</li> <li>• <b>&lt;Amount&gt;Amount&lt;/Amount&gt;</b> Total amount to search</li> </ul>   |

|  |   |
|--|---|
|  | <p>transactions for in DDDD.CC format.</p> <ul style="list-style-type: none"> <li>• <b>&lt;RegisterNum&gt;</b><i>RegisterNum</i><b>&lt;/RegisterNum&gt;</b> Register number, originally passed with the transaction, to search transactions for.</li> </ul> |
|--|---|

## Example

The following table contains sample data you can use to test this Web service. The **UserName**, **Password**, and **RPNum** parameters should be changed when testing this example yourself. The example data shown will create a list of sale card transactions that were processed between 1/1/2000 and 1/1/3000. The format of the output will show descriptive headers and will be similar to a Comma Separate Values (CSV) format because of the use of the "," character as the column delimiter.

| Parameter            | Value    |
|----------------------|----------|
| <b>UserName</b>      | test     |
| <b>Password</b>      | 123      |
| <b>RPNum</b>         | 1        |
| <b>BeginDt</b>       | 1/1/2000 |
| <b>EndDt</b>         | 1/1/3000 |
| <b>TransType</b>     | 'Sale'   |
| <b>ExcludeVoid</b>   | TRUE     |
| <b>TransformType</b> | DELIM    |
| <b>ColDelim</b>      | ,        |
| <b>RowDelim</b>      |          |
| <b>IncludeHeader</b> | TRUE     |

Result: The following is the result of the using the Web Services with the sample data from the table above.

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<string xmlns="http://localhost/admin/ws ">
```

```
TRX_HD_Key,Invoice_ID,Seq_Num_CH,Date_DT,Merchant_Key,User_Name
_VC,Register_Number_CH,Reseller_Key,Payment_Type_ID,Trans_Type_ID,
Processor_ID,TRX_Settle_Key,TRX_Settle_Msg_VC,Void_Flag_CH,Settle_Fla
g_CH,Ref_Number_CH,Settle_Date_DT,Last_Update_DT,TRX_Card_Key,Card
_Info_Key,Auth_Amt_MN,Tip_Amt_MN,Total_Amt_MN,Cash_Back_Amt_MN,
SureCharge_Amt_MN,Account_Type_CH,Result_CH,Result_Txt_VC,Approval
_Code_CH,Host_Ref_Num_CH,AVS_Resp_CH,AVS_Resp_Txt_VC,CV_Resp_C
H,CV_Resp_Txt_VC,Host_Date_CH,Host_Time_CH,Acct_Num_CH,Exp_CH,Ty
pe_CH,Name_on_Card_VC,Street_CH,Zip_CH,Track_VC,Pin_Block_CH,TRX_
```

```

Receipt_key,Create_Date_DT,Receipt_Type_ID,IP_VC|14,,,12/3/2003
10:47:48 AM,1,test , ,100,VISA ,Sale ,VITAL ,14,GB00004
ACCEPTED,,1,,12/3/2003 10:46:34 AM,12/3/2003 10:48:13
AM,14,14,2.5000,0,2.5000,0,0,VISA ,0 ,APPROVAL VITAL1 ,VITAL1 , ,0,,
,,,,4266503700000247,1009 ,VISA ,VINCE VISA , , ,1,
,,,,127.0.0.1|13,,,12/3/2003 10:04:29 AM,1,test , ,100,MASTERCARD,Sale
,VITAL ,13,GB00004 ACCEPTED,,1,,12/3/2003 10:46:34 AM,12/3/2003
10:48:13 AM,13,13,2.8500,0,2.8500,0,0,MASTERCARD,0 ,APPROVAL VITAL9
,VITAL9 , ,0,, ,,,,,542400000000015,0905 ,MASTERCARD, , , ,
,,,,127.0.0.1|12,,,12/3/2003 9:45:59 AM,1,test , ,100,VISA ,Sale ,VITAL
,12,GB00003 ACCEPTED,,1,,12/3/2003 9:46:16 AM,12/3/2003 9:46:16
AM,12,12,1.6800,0,1.6800,0,0,VISA ,0 ,APPROVAL VITAL3 ,VITAL3 , ,0,,
,,,,4126196901499,0905 ,VISA , , , , ,,,,,127.0.0.1|10,,,12/3/2003 9:35:19
AM,1,test , ,100,VISA ,Sale ,VITAL ,10,GB00003 ACCEPTED,,1,,12/3/2003
9:46:16 AM,12/3/2003 9:46:16 AM,10,10,2.3500,0,2.3500,0,0,VISA ,0
,APPROVAL VITAL8 ,VITAL8 , ,0,, ,,,,,4055016727870315,0905 ,VISA , , , ,
,,,,127.0.0.1|7,,,12/3/2003 9:27:57 AM,1,test , ,100,VISA ,Sale ,VITAL
,7,GB00002 ACCEPTED,,1,,12/3/2003 9:26:35 AM,12/3/2003 9:28:16
AM,7,7,1.2500,0,1.2500,0,0,VISA ,0 ,APPROVAL VITAL4 ,VITAL4 , ,0,,
,,,,4007000000027,0905 ,VISA , , , , ,,,,,127.0.0.1|6,,,12/3/2003 9:27:44
AM,1,test , ,100,VISA ,Sale ,VITAL ,6,GB00002 ACCEPTED,,1,,12/3/2003
9:26:35 AM,12/3/2003 9:28:16 AM,6,6,2.0000,0,2.0000,0,0,VISA ,0
,APPROVAL VITAL9 ,VITAL9 , ,0,, ,,,,,4007000000027,0905 ,VISA , , , ,
,,,,127.0.0.1|5,,,12/3/2003 9:27:15 AM,1,test , ,100,MASTERCARD,Sale
,VITAL ,5,,,,,12/3/2003 9:27:17
AM,5,5,1.2500,0,1.2500,0,0,MASTERCARD,12 , ACCT LENGTH ERR,EA , ,0,,
,,,,5405001478615777532,0905 ,MASTERCARD, , , ,
,,,,127.0.0.1|2,,,12/3/2003 9:03:16 AM,1,test , ,100,MASTERCARD,Sale
,VITAL ,2,GB00001 ACCEPTED,,1,,12/3/2003 9:03:40 AM,12/3/2003
9:03:40 AM,2,2,2.4500,0,2.4500,0,0,MASTERCARD,0 ,APPROVAL VITAL7
,VITAL7 , ,0,, ,,,,,542400000000015,0905 ,MASTERCARD, , , ,
,,,,127.0.0.1|1,,,12/3/2003 8:58:02 AM,1,test , ,100,VISA ,Sale ,VITAL
,1,GB00001 ACCEPTED,,1,,12/3/2003 9:03:40 AM,12/3/2003 9:03:40
AM,1,1,1.0000,0,1.0000,0,0,VISA ,0 ,APPROVAL VITAL3 ,VITAL3 , ,0,,
,,,,4111111111111111,0905 ,VISA , , , , ,,,,,127.0.0.1

```

</string>

## GetCardTrxSummary

This Web service operation retrieves a card transaction summary for a merchant. The URL to access this Web service is:  
<https://localhost/admin/ws/trxdetail.aspx?op=GetCardTrxSummary>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter       | Value   |
|-----------------|---|
| <b>UserName</b> | Required. User name assigned in the payment server                  |
| <b>Password</b> | Required. Password for the user name assigned in the payment server |

|                      |   |
|----------------------|---|
| <b>RPNum</b>         | Required. The merchant's RP Number in order to uniquely identify the merchant's account for the query   |
| <b>BeginDt</b>       | Required. The begin date of the date range in MM/DD/YYYY format. This date will be converted to MM/DD/YYYYT00:00:00:0000AM  |
| <b>EndDt</b>         | Required. The end date of the date range in MM/DD/YYYY format. This date will be converted to MM/DD/YYYYT12:59:59:9999PM  |
| <b>ApprovalCode</b>  | Optional. If provided, only those transactions matching the <b>ApprovalCode</b> parameter will be included  |
| <b>Register</b>      | Optional. The register that originated the transaction. If provided, only those transactions with the matching register will be included  |
| <b>NameOnCard</b>    | Optional. Cardholder's name as it appears on the card. If provided, only those transactions with cardholder's name matching <b>NameOnCard</b> will be included. Matching is done using wild cards: e.g. "test" will match "test", "1test" and "1test234"  |
| <b>CardNum</b>       | Optional. A card number. If provided, only those transactions with the cardholder's name matching <b>CardNum</b> will be included. Matching is done using wild cards  |
| <b>CardType</b>      | Optional. If provided, only those transactions matching the <b>CardType</b> will be included. Valid values are: <ul style="list-style-type: none"> <li>• <b>'AMEX'</b> American Express card</li> <li>• <b>'CARTBLANCH'</b> Carte Blanch card</li> <li>• <b>'DEBIT'</b> Debit card</li> <li>• <b>'DINERS'</b> Diners Club card</li> <li>• <b>'DISCOVER'</b> Novus Discover card</li> <li>• <b>'EBT'</b> Electronic Benefit Transfer</li> <li>• <b>'JAL'</b> JAL card</li> <li>• <b>'JCB'</b> Japanese Commercial Bank card</li> <li>• <b>'MASTERCARD'</b> Master card</li> <li>• <b>'VISA'</b> Visa card</li> <li>• <b>'EGC'</b> Gift card</li> </ul> |
| <b>ExcludeVoid</b>   | Required. Whether to exclude voided transactions; must either be <b>TRUE</b> or <b>FALSE</b> . Default is <b>TRUE</b>   |
| <b>User</b>          | Optional. The user who originated the transactions. If provided, only those transactions created by the matching <b>User</b> will be included. Matching is done using wild cards  |
| <b>SettleFlag</b>    | Optional. An option to retrieve the settled transactions or unsettled transactions; must either be <b>TRUE</b> or <b>FALSE</b>  |
| <b>SettleMsg</b>     | Optional. The settlement ID or message returned from the host   |
| <b>SettleDt</b>      | Optional. The settlement timestamp  |
| <b>TransformType</b> | Optional. The type of format to transform the data into. Leave the field blank to default to <b>XML</b> <ul style="list-style-type: none"> <li>• <b>XML</b> will output the plain XML string</li> </ul>   |

|                      |   |
|----------------------|---|
|                      | <ul style="list-style-type: none"> <li>• <b>XSL</b> will use XSL to transform the XML output</li> <li>• <b>DELIM</b> uses <b>ColDelim</b> and <b>RowDelim</b> to format the output</li> </ul>   |
| <b>Xsl</b>           | <p>Optional. This field is used only if the <b>TransformType</b> is <b>XSL</b>. The XSL to transform the resulting dataset; if provided, the resulting dataset will be transformed using this XSL. You may pass in a URL to the XSL file, or the XSL string itself. If this field is not empty, the Web Services will try to locate the file from the URL. If that also fails, it will treat it as a XSL string. In any case, the final XSL string will be loaded and validated against the XSL schema; if it passes, then that XSL will be used for transformation. A sample predefined XSL is included with this Web Services:</p> <ul style="list-style-type: none"> <li>• <b>http://localhost/admin/ws/TabDelim.xsl</b> for a tab delimited transformation</li> </ul> |
| <b>ColDelim</b>      | Optional. This field is used only if the <b>TransformType</b> is <b>DELIM</b> . This defines the string that separates each column  |
| <b>RowDelim</b>      | Optional. This field is used only if the <b>TransformType</b> is <b>DELIM</b> . This defines the string that separates each row   |
| <b>IncludeHeader</b> | Optional. This field is used only if the <b>TransformType</b> is <b>DELIM</b> . If <b>TRUE</b> , then field headers will be included in the first row using the same delimiter strings; must either be <b>TRUE</b> or <b>FALSE</b>  |
| <b>ExtData</b>       | <p>Optional. Extended data in XML format. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>&lt;IMAGE_TYPE&gt;NO_IMAGE&lt;/IMAGE_TYPE&gt;</b> for no image</li> <li>• <b>&lt;IMAGE_TYPE&gt;ONLY_IMAGE&lt;/IMAGE_TYPE&gt;</b> for only the image</li> <li>• <b>&lt;IMAGE_TYPE&gt;ALL&lt;/IMAGE_TYPE&gt;</b> for all images</li> </ul>  |

## Example

The following table contains sample data you can use to test this Web service. The **UserName**, **Password**, and **RPNum** parameters should be changed when testing this example yourself. The example data shown will create a summarized list of Visa card transactions that were processed between 1/1/2000 and 1/1/3000. The output data is in XML format because the **TransformType** parameter was not specified.

| Parameter          | Value    |
|--------------------|----------|
| <b>UserName</b>    | Test     |
| <b>Password</b>    | 123      |
| <b>RPNum</b>       | 1        |
| <b>BeginDt</b>     | 1/1/2000 |
| <b>EndDt</b>       | 1/1/3000 |
| <b>CardType</b>    | VISA     |
| <b>ExcludeVoid</b> | FALSE    |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
<string xmlns="http://localhost/Admin/ws">
  <CardTrxSummary> <PaymentMethod> <Payment_Type_ID>VISA
  </Payment_Type_ID> <Authorization>1.0000</Authorization>
  <Capture>0</Capture> <ForceCapture>0</ForceCapture>
  <PostAuth>0</PostAuth> <Return>0</Return> <Sale>2.0000</Sale>
  <Receipt>0</Receipt> <RepeatSale>0</RepeatSale>
  <Activate>0</Activate> <Deactivate>0</Deactivate>
  <Reload>0</Reload> <Authorization_Cnt>1</Authorization_Cnt>
  <Capture_Cnt>0</Capture_Cnt>
  <ForceCapture_Cnt>0</ForceCapture_Cnt>
  <PostAuth_Cnt>0</PostAuth_Cnt> <Return_Cnt>0</Return_Cnt>
  <Sale_Cnt>2</Sale_Cnt> <Receipt_Cnt>0</Receipt_Cnt>
  <RepeatSale_Cnt>0</RepeatSale_Cnt> <Activate_Cnt>0</Activate_Cnt>
  <Deactivate_Cnt>0</Deactivate_Cnt> <Reload_Cnt>0</Reload_Cnt>
  <Cnt>3</Cnt> </PaymentMethod> </CardTrxSummary>
</string>
```

## GetCheckTrx

This Web service operation retrieves checks transaction details for a merchant. The URL to access this Web service is:

<https://localhost/admin/ws/trxdetail.asmx?op=GetCheckTrx>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter          | Value   |
|--------------------|---|
| <b>UserName</b>    | Required. User name assigned in the payment server  |
| <b>Password</b>    | Required. Password for the user name assigned in the payment server   |
| <b>RPNum</b>       | Required. The merchant's RP Number in order to uniquely identify merchant's account for the query   |
| <b>PNRef</b>       | Optional. The unique payment reference number assigned to the transaction. If this field is provided, all other query fields will be ignored when using <b>PNRef</b> parameter to query the system  |
| <b>BeginDt</b>     | Required, except when <b>PNRef</b> is provided. The begin date of the date range in MM/DD/YYYY (or YYYY-MM-DD or YYYY-MM-DDThh:mm:ss) format. This date will be converted to YYYY-MM-DDThh:mm:ss (time is in 24-hour format). If the passed-in BeginDt does not contain time information, BeginDt will default to midnight on the given date. For example:<br>2005-08-19T12:00:12 is kept as is<br>2005-08-19 becomes 2005-08-19T00:00:00<br>2005/08/19 becomes 2005-08-19T00:00:00<br>08/19/2005 becomes 2005-08-19T00:00:00<br>The query to obtain transactions in the date range is constructed as follows:<br>(Date DT >=BeginDt) AND (Date DT<EndDt)<br>where Date DT is the transaction timestamp |
| <b>EndDt</b>       | Required, except when <b>PNRef</b> is provided. The end date of the date range in MM/DD/YYYY (or YYYY-MM-DD or YYYY-MM-DDThh:mm:ss) format. This date will be converted to YYYY-MM-DDThh:mm:ss (time is in 24-hour format). If the passed-in EndDt does not contain time information, EndDt will be incremented to the <i>next day</i> at midnight such that no transaction on the desired end date will be excluded based on its time. For example:<br>2005-08-19T12:00:12 is kept as is<br>2005-08-19 becomes 2005-08-20T00:00:00<br>08/19/2005 becomes 2005-08-20T00:00:00   |
| <b>PaymentType</b> | Optional. If provided, only those transactions matching the <b>PaymentType</b> will be included. Valid values are: <ul style="list-style-type: none"> <li>• <b>'ACH'</b> Automated Clearing House</li> <li>• <b>'ECHECK'</b> Electronic Check</li> <li>• <b>'GUARANTEE'</b> Guarantee check</li> <li>• <b>'PAYRECEIPT'</b> to retrieve receipt images that were uploaded to the payment server</li> <li>• <b>'SETTLE'</b> to retrieve requests to settle transactions</li> </ul>  |

|                           |   |
|---------------------------|---|
|                           | <ul style="list-style-type: none"> <li>• <b>'VERIFY'</b> to retrieve pre-authorized checks</li> </ul> <p>Or any permutation of the above values, e.g. <b>"ACH', 'ECHECK"</b> will pull all transactions with either ACH or ECHECK payment types</p>   |
| <b>ExcludePaymentType</b> | Optional. If provided, any transaction matching the <b>ExcludePaymentType</b> will be excluded  |
| <b>TransType</b>          | <p>Optional. If provided, only those transactions matching the <b>TransType</b> will be included. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>'Authorization'</b> to retrieve previously-authorized (pre-auth) transactions</li> <li>• <b>'Capture'</b> to retrieve captured transactions</li> <li>• <b>'Credit'</b> to retrieve return transactions</li> <li>• <b>'ForceCapture'</b> to retrieve force-auth transactions</li> <li>• <b>'GetStatus'</b> to make an inquiry to the EBT or gift card's balance</li> <li>• <b>'PostAuth'</b> to retrieve post-auth transactions</li> <li>• <b>'Purged'</b> to remove a transaction from the current batch due to an error</li> <li>• <b>'Receipt'</b> to retrieve receipt images that were uploaded to the payment server</li> <li>• <b>'RepeatSale'</b> to retrieve repeat-sale transactions</li> <li>• <b>'Sale'</b> to retrieve sale transactions</li> <li>• <b>'Void'</b> to retrieve void transactions</li> </ul> <p>Or any permutation of the above values, e.g. <b>"Credit', 'Sale"</b> will pull all transactions with either Credit or Sale transaction types</p> |
| <b>ExcludeTransType</b>   | Optional. If provided, any transaction matching the <b>ExcludeTransType</b> will be excluded  |
| <b>ApprovalCode</b>       | Optional. If provided, only those transactions matching the <b>ApprovalCode</b> parameter will be included  |
| <b>Result</b>             | <p>Optional. If provided, only those transactions matching the <b>Result</b> will be included. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>0</b> is Approved</li> <li>• Anything else is declined, if you want all the declined transactions, you should leave this field empty and use the <b>ExcludeResult=0</b> instead</li> </ul>   |
| <b>ExcludeResult</b>      | Optional. If provided, any transactions matching the <b>ExcludedResult</b> will be excluded   |
| <b>NameOnCheck</b>        | Optional. Check owner's name as it appear on the check, if provided. Only those transactions with check owner's name matching <b>NameOnCheck</b> will be included. Matching is done using wild cards: e.g. "test" will match "test", "1test" and "1test234"   |
| <b>CheckNum</b>           | Optional. Check number. If provided, only those transactions with matching <b>CheckNum</b> will be included   |

|                      |  |
|----------------------|--|
| <b>AcctNum</b>       | Optional. Check account number. If provided, only those transactions matching the <b>AcctNum</b> will be included. Matching is done using wild cards   |
| <b>RouteNum</b>      | Optional. If provided, any transactions matching the <b>RouteNum</b> (Transit Number) will be excluded. Matching is done using wild cards  |
| <b>ExcludeVoid</b>   | Optional. Whether to exclude voided transactions; must either be <b>TRUE</b> or <b>FALSE</b> . The default value is <b>TRUE</b>  |
| <b>User</b>          | Optional. The user who originated the transactions. If provided, only those transactions created by the matching <b>User</b> will be included. Matching is done using wild cards   |
| <b>InvoiceId</b>     | Optional. The invoice ID that was included in the original transaction. If provided, only those transactions with matching <b>invoiceId</b> will be included. Matching is done using wild cards  |
| <b>SettleFlag</b>    | Optional. Whether the transaction was settled; must either be <b>1</b> for true or <b>0</b> for false  |
| <b>SettleMsg</b>     | Optional. The settlement ID or message returned from the host  |
| <b>SettleDt</b>      | Optional. The settlement timestamp   |
| <b>TransformType</b> | Optional. The type of format to transform the data into. Leave the field blank to default to <b>XML</b> <ul style="list-style-type: none"> <li>• <b>XML</b> will output the plain XML string</li> <li>• <b>XSL</b> will use XSL to transform the XML output</li> <li>• <b>DELIM</b> uses <b>ColDelim</b> and <b>RowDelim</b> to format the output</li> </ul>   |
| <b>Xsl</b>           | Optional. This field is used only if the <b>TransformType</b> is <b>XSL</b> . The XSL to transform the resulting dataset; if provided, the resulting dataset will be transformed using this XSL. You may pass in a URL to the XSL file, or the XSL string itself. If this field is not empty, the Web Services will try to locate the file from the URL. If that also fails, it will treat it as a XSL string. In any case, the final XSL string will be loaded and validated against the XSL schema; if it passes, then that XSL will be used for transformation. A sample predefined XSL is included with this Web Services: <ul style="list-style-type: none"> <li>• <b>http://localhost/admin/ws/TabDelim.xsl</b> for a tab delimited transformation.</li> </ul> |
| <b>ColDelim</b>      | Optional. This field is used only if the <b>TransformType</b> is <b>DELIM</b> . This defines the string that separates each column   |
| <b>RowDelim</b>      | Optional. This field is used only if the <b>TransformType</b> is <b>DELIM</b> . This defines the string that separates each row  |
| <b>IncludeHeader</b> | Optional. This field is used only if the <b>TransformType</b> is <b>DELIM</b> . If <b>TRUE</b> , then field headers will be included in the first row using the same delimiter strings; must either be <b>TRUE</b> or <b>FALSE</b>   |
| <b>ExtData</b>       | Optional. Extended data in XML format. Valid values are: <ul style="list-style-type: none"> <li>• <b>&lt;IMAGE_TYPE&gt;NO_IMAGE&lt;/IMAGE_TYPE&gt;</b> for no image</li> <li>• <b>&lt;IMAGE_TYPE&gt;ONLY_IMAGE&lt;/IMAGE_TYPE&gt;</b> for only the image</li> <li>• <b>&lt;IMAGE_TYPE&gt;ALL&lt;/IMAGE_TYPE&gt;</b> for all images</li> </ul>  |

|  |  |
|--|--|
|  | <ul style="list-style-type: none"> <li>• <b>&lt;CustomerID&gt;</b><i>CustomerID</i><b>&lt;/CustomerID&gt;</b> for customer ID</li> <li>• <b>&lt;Amount&gt;</b><i>Amount</i><b>&lt;/Amount&gt;</b> Total amount to search transactions for in DDDD.CC format.</li> <li>• <b>&lt;RegisterNum&gt;</b><i>RegisterNum</i><b>&lt;/RegisterNum&gt;</b> Register number, originally passed with the transaction, to search transactions for</li> </ul> |
|--|--|

## Example

The following table contains sample data you can use to test this Web service. The **UserName**, **Password**, and **RPNum** parameters should be changed when testing this example yourself. The example data shown will create a list of approved Verify check transactions that were processed between 1/1/2003 and 12/31/2003. The output data is in XML format because the **TransformType** parameter was not specified.

| Parameter          | Value      |
|--------------------|------------|
| <b>UserName</b>    | Test       |
| <b>Password</b>    | 123        |
| <b>RPNum</b>       | 1          |
| <b>BeginDt</b>     | 1/1/2003   |
| <b>EndDt</b>       | 12/31/2003 |
| <b>PaymentType</b> | 'VERIFY'   |
| <b>Result</b>      | 0          |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<string xmlns=" http://localhost/admin/ws">
```

```

<RichDBDS> <TrxDetailCheck> <RichDBDS> <TrxDetailCheck>
<TRX_HD_Key>25</TRX_HD_Key> <Invoice_ID /> <Date_DT>2003-12-
05T13:47:15.1230000-08:00</Date_DT>
<Merchant_Key>1</Merchant_Key> <User_Name_VC>test
</User_Name_VC> <Register_Number_CH> </Register_Number_CH>
<Reseller_Key>100</Reseller_Key> <Payment_Type_ID>VERIFY
</Payment_Type_ID> <Trans_Type_ID>Authorization </Trans_Type_ID>
<Processor_ID>RMRS </Processor_ID> <Last_Update_DT>2003-12-
05T13:47:15.1230000-08:00</Last_Update_DT>
<TRX_Check_Key>5</TRX_Check_Key> <CheckNum_CH>1001
</CheckNum_CH> <MICR_VC />
<AccountNum_VC>*****7890</AccountNum_VC>
<TransitNum_VC>123456780</TransitNum_VC> <RawMICR_VC />
<DL_VC /> <SS_CH /> <DOB_CH> </DOB_CH> <StateCode_CH>DE

```

```

</StateCode_CH> <NameOnCheck_VC>Jane Doe</NameOnCheck_VC>
<EMail_VC /> <Phone_VC /> <Amount_MN>1.0000</Amount_MN>
<Result_CH>0 </Result_CH> <Result_Txt_VC>&lt;INPUT TYPE=HIDDEN
NAME=PARAM_RESPONSE_CODE VALUE="AA"&gt;&lt;INPUT TYPE=HIDDEN
NAME=PARAM_AUTH_SOURCE VALUE="0"&gt;&lt;INPUT TYPE=HIDDEN
NAME=PARAM_CAP_CODE VALU</Result_Txt_VC>
<Host_Approval_CH>963852 </Host_Approval_CH> <Host_Ref_Num_CH>
</Host_Ref_Num_CH> <Result_Msg_VC>APPROVAL
963852</Result_Msg_VC> <Result_Msg1_VC /> <Result_Msg2_VC />
<Host_Date_CH>120503 </Host_Date_CH> <Host_Time_CH>164502
</Host_Time_CH> <IP_VC>192.168.2.22</IP_VC> </TrxDetailCheck>
<TrxDetailCheck> <TRX_HD_Key>24</TRX_HD_Key> <Invoice_ID />
<Date_DT>2003-12-05T13:46:50.0770000-08:00</Date_DT>
<Merchant_Key>1</Merchant_Key> <User_Name_VC>test
</User_Name_VC> <Register_Number_CH> </Register_Number_CH>
<Reseller_Key>100</Reseller_Key> <Payment_Type_ID>VERIFY
</Payment_Type_ID> <Trans_Type_ID>Authorization </Trans_Type_ID>
<Processor_ID>RMRS </Processor_ID> <Last_Update_DT>2003-12-
05T13:46:50.0770000-08:00</Last_Update_DT>
<TRX_Check_Key>4</TRX_Check_Key> <CheckNum_CH>4877
</CheckNum_CH> <MICR_VC />
<AccountNum_VC>*****7890</AccountNum_VC>
<TransitNum_VC>123456780</TransitNum_VC> <RawMICR_VC />
<DL_VC /> <SS_CH /> <DOB_CH> </DOB_CH> <StateCode_CH>WA
</StateCode_CH> <NameOnCheck_VC>John Doe</NameOnCheck_VC>
<EMail_VC /> <Phone_VC /> <Amount_MN>2.0000</Amount_MN>
<Result_CH>0 </Result_CH> <Result_Txt_VC>&lt;INPUT TYPE=HIDDEN
NAME=PARAM_RESPONSE_CODE VALUE="AA"&gt;&lt;INPUT TYPE=HIDDEN
NAME=PARAM_AUTH_SOURCE VALUE="0"&gt;&lt;INPUT TYPE=HIDDEN
NAME=PARAM_CAP_CODE VALU</Result_Txt_VC>
<Host_Approval_CH>963852 </Host_Approval_CH> <Host_Ref_Num_CH>
</Host_Ref_Num_CH> <Result_Msg_VC>APPROVAL
963852</Result_Msg_VC> <Result_Msg1_VC /> <Result_Msg2_VC />
<Host_Date_CH>120503 </Host_Date_CH> <Host_Time_CH>164437
</Host_Time_CH> <IP_VC>192.168.2.22</IP_VC> </TrxDetailCheck>
</RichDBDS>

```

</string>

## GetCardType

This Web service operation returns the name of the card issuer, such as Visa, MasterCard, Amex. The URL to access this operation is: <https://localhost/SmartPayments/validate.asmx?op=GetCardType>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter         | Description                           |
|-------------------|---------------------------------------|
| <b>CardNumber</b> | Required. The number of a credit card |

## Example

The following table contains sample data you can use to test this Web service.

| Parameter         | Value            |
|-------------------|------------------|
| <b>CardNumber</b> | 5454545454545454 |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<string xmlns="http://localhost/SmartPayments/">MASTERCARD</string>
```

## GetInfo

This Web service retrieves information pertaining to the transaction type (TransType) specified. The URL to access this Web service is:  
<https://localhost/SmartPayments/transact.asmx?op=GetInfo>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter        | Description   |
|------------------|---|
| <b>UserName</b>  | Required. User name assigned in the payment server  |
| <b>Password</b>  | Required. Password for the user name assigned in the payment server   |
| <b>TransType</b> | Required. Valid values are: <ul style="list-style-type: none"><li>• <b>BatchInquiry</b> returns a comma delimited list in a single XML tag that contains the summarized transaction dollar amount and transaction count for each payment method in the current batch. The list is in the following format:<br/><i>Payment_Method1=0.00, Payment_Method2=0.00</i></li><li>• <b>Setup</b> returns a comma delimited list in a single XML tag that contains merchant setup information. The list is in the following format:<br/><i>Setup_Name1=Y N, Setup_Name2=Y N</i></li><li>• <b>StatusCheck</b> returns "OK" if a connection can be made to the payment server with the supplied user name and password; otherwise, an error message is returned</li><li>• <b>Initialize</b> returns the merchant account setup, including Partner number, Merchant ID, credit card type, phone number, etc.</li></ul> |
| <b>ExtData</b>   | Optional. Extended data in XML format. Valid values are: <ul style="list-style-type: none"><li>• <b>&lt;TrainingMode&gt;T&lt;/TrainingMode&gt;</b> an indicator that specifies transactions will be processed for local loop back testing</li><li>• <b>&lt;TrainingMode&gt;F&lt;/TrainingMode&gt;</b> an indicator that specifies transactions will not be processed for local loop back testing</li><li>• <b>&lt;BatchSequenceNum&gt;Number&lt;/BatchSequenceNum&gt;</b> used when the <b>TransType</b> is <b>BatchInquiry</b> and it is a number that indicates which previous batch</li></ul>  |

|  |  |
|--|--|
|  | <p>or current batch the Payment Server should query from the processor in order to get information about the batch. Currently only support with Global Payments. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>0</b> = Current open batch (default value if <b>BatchSequenceNum</b> is not specified in <b>ExtData</b>)</li> <li>• <b>1</b> = previous batch</li> <li>• <b>2</b> = the batch before the previous batch specified with the value <b>1</b></li> <li>• <i>N</i> = and so on...</li> </ul> |
|--|--|

## Examples

The following examples show the results of testing four **TransTypes**. When testing each example yourself, the **UserName** and **Password** parameters should be changed.

### Example 1: BatchInquiry

| Parameter        | Value        |
|------------------|--------------|
| <b>UserName</b>  | Test         |
| <b>Password</b>  | 123          |
| <b>TransType</b> | BatchInquiry |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://localhost/SmartPayments/"><Result>0</Result><RespMSG>A
pproved</RespMSG><ExtData>Credit_Sale_Amount=28.00,Credit_Sale_Coun
t=28,Credit_Return_Amount=7.00,Credit_Return_Count=7,Credit_Net_Amo
unt=21.00,Credit_Net_Count=35,Debit_Sale_Amount=9.00,Debit_Sale_Cou
nt=6,Debit_Return_Amount=1.00,Debit_Return_Count=1,Debit_Net_Amou
nt=8.00,Debit_Net_Count=7,Check_Sale_Amount=0.00,Check_Sale_Count=
0,Check_Net_Amount=0.00,Check_Net_Count=0</ExtData></Response>
```

### Example 2: Setup

| Parameter        | Value |
|------------------|-------|
| <b>UserName</b>  | Test  |
| <b>Password</b>  | 123   |
| <b>TransType</b> | Setup |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://localhost/SmartPayments/"><Result>0</Result><RespMSG>A
pproved</RespMSG><ExtData>Force_Duplicates=N,Auto_Close_Batch=Y,Ind
ustry=R,DEBIT=Y,AMEX=Y,CARTBLANCH=Y,DINERS=Y,DISCOVER=Y,JAL=Y,
JCB=Y,MASTERCARD=Y,VISA=Y,EBT=Y</ExtData></Response>
```

### Example 3: StatusCheck

| Parameter        | Value       |
|------------------|-------------|
| <b>UserName</b>  | Test        |
| <b>Password</b>  | 123         |
| <b>TransType</b> | StatusCheck |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://localhost/SmartPayments/"><Result>0</Result><RespMSG>A
pproved</RespMSG><ExtData>OK</ExtData></Response>
```

### Example 4: Initialize

| Parameter        | Value      |
|------------------|------------|
| <b>UserName</b>  | Test       |
| <b>Password</b>  | 123        |
| <b>TransType</b> | Initialize |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><ExtData><Partner>110<
/Partner> <Vendor>206</Vendor> <MerchantID /> <PinPadKeyManagement
/><LiveURL>https://www.YourWebSite.com</LiveURL><LiveURL1>https://w
ww1.YourWebSite.com</LiveURL1><TestURL>https://test.YourWebSite.com
</TestURL><TestURL1>https://test.YourWebSite.com</TestURL1><EPSPay>/
pay/payxml.aspx</EPSPay><EPSlogin>/Admin/login.aspx</EPSlogin><Phone
1>425-123-1234</Phone1><Phone2>425-123-
1234</Phone2><Auto_Close_Batch>N</Auto_Close_Batch><eCheck>Y</eCheck>
<CreditCard>Y</CreditCard><PaymentTypes><CardType>Amex</CardType><Car
dType>CartBlanch</CardType><CardType>Diners</CardType><CardType>Disc
```

```

over</CardType><CardType>JAL</CardType><CardType>JCB</CardType><Card
Type>MasterCard</CardType><CardType>Visa</CardType></PaymentTypes><E
xpressPay><CardType>Visa</CardType><LTAmount>25</LTAmount><Am
ount>0</Amount><EntryMethod>S</EntryMethod><ProcessingRule>On-
Line</ProcessingRule><PrintReceipt>N</PrintReceipt></ExpressPay></Ext
Data>

```

```
</Response>
```

## GetOpenBatchSummary

This Web service operation retrieves a payment type transaction summary of the current open batch for a merchant. The URL to access this Web service is:

<https://localhost/admin/ws/trxdetail.aspx?op=GetOpenBatchSummary>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter       | Value   |
|-----------------|---|
| <b>UserName</b> | Required. User name assigned in the payment server  |
| <b>Password</b> | Required. Password for the user name assigned in the payment server   |
| <b>RPNum</b>    | Required. The number uniquely identifies each merchant  |
| <b>BeginDt</b>  | Optional. The begin date of the date range in MM/DD/YYYY format. This date will be converted to: MM/DD/YYYYT00:00:00:0000AM |
| <b>EndDt</b>    | Optional. The end date of the date range in MM/DD/YYYY format. This date will be converted to: MM/DD/YYYYT12:59:59:9999PM   |
| <b>ExtData</b>  | Currently there is no data value available  |

## Example

The following table contains sample data you can use to test this Web service. The **UserName**, **Password**, and **RPNum** parameters should be changed when testing this example yourself. The example data shown will create a categorized summary list of all transactions in the current open batch.

| Parameter       | Value |
|-----------------|-------|
| <b>UserName</b> | Test  |
| <b>Password</b> | 123   |
| <b>RPNum</b>    | 1     |

Result:

<?xml version="1.0" encoding="utf-8" ?>

```
<string xmlns="http://localhost/Admin/ws"><OpenBatchSummary> <Table>
  <Payment_Type_ID>DEBIT </Payment_Type_ID>
  <Authorization>0</Authorization> <Capture>0</Capture>
  <ForceCapture>0</ForceCapture> <PostAuth>0</PostAuth>
  <Return>7.0000</Return> <Sale>61.0000</Sale>
  <Receipt>0</Receipt> <RepeatSale>0</RepeatSale>
  <Authorization_Cnt>0</Authorization_Cnt>
  <Capture_Cnt>0</Capture_Cnt>
  <ForceCapture_Cnt>0</ForceCapture_Cnt>
  <PostAuth_Cnt>0</PostAuth_Cnt> <Return_Cnt>7</Return_Cnt>
  <Sale_Cnt>58</Sale_Cnt> <Receipt_Cnt>0</Receipt_Cnt>
  <RepeatSale_Cnt>0</RepeatSale_Cnt> <Cnt>65</Cnt> </Table>
  <Table> <Payment_Type_ID>EBT </Payment_Type_ID>
  <Authorization>0</Authorization> <Capture>0</Capture>
  <ForceCapture>0</ForceCapture> <PostAuth>0</PostAuth>
  <Return>132.0000</Return> <Sale>150.0000</Sale>
  <Receipt>0</Receipt> <RepeatSale>0</RepeatSale>
  <Authorization_Cnt>0</Authorization_Cnt>
  <Capture_Cnt>0</Capture_Cnt>
  <ForceCapture_Cnt>0</ForceCapture_Cnt>
  <PostAuth_Cnt>0</PostAuth_Cnt> <Return_Cnt>15</Return_Cnt>
  <Sale_Cnt>24</Sale_Cnt> <Receipt_Cnt>0</Receipt_Cnt>
  <RepeatSale_Cnt>0</RepeatSale_Cnt> <Cnt>39</Cnt> </Table>
  <Table> <Payment_Type_ID>MASTERCARD</Payment_Type_ID>
  <Authorization>0</Authorization> <Capture>0</Capture>
  <ForceCapture>0</ForceCapture> <PostAuth>0</PostAuth>
  <Return>0</Return> <Sale>4.0000</Sale> <Receipt>0</Receipt>
  <RepeatSale>7.0000</RepeatSale>
  <Authorization_Cnt>0</Authorization_Cnt>
  <Capture_Cnt>0</Capture_Cnt>
  <ForceCapture_Cnt>0</ForceCapture_Cnt>
  <PostAuth_Cnt>0</PostAuth_Cnt> <Return_Cnt>0</Return_Cnt>
  <Sale_Cnt>4</Sale_Cnt> <Receipt_Cnt>0</Receipt_Cnt>
  <RepeatSale_Cnt>7</RepeatSale_Cnt> <Cnt>11</Cnt> </Table>
  <Table> <Payment_Type_ID>VISA </Payment_Type_ID>
  <Authorization>0</Authorization> <Capture>0</Capture>
  <ForceCapture>1.0000</ForceCapture> <PostAuth>1.0000</PostAuth>
  <Return>0</Return> <Sale>19.0000</Sale> <Receipt>0</Receipt>
  <RepeatSale>0</RepeatSale>
  <Authorization_Cnt>0</Authorization_Cnt>
  <Capture_Cnt>0</Capture_Cnt>
  <ForceCapture_Cnt>1</ForceCapture_Cnt>
  <PostAuth_Cnt>1</PostAuth_Cnt> <Return_Cnt>0</Return_Cnt>
  <Sale_Cnt>19</Sale_Cnt> <Receipt_Cnt>0</Receipt_Cnt>
  <RepeatSale_Cnt>0</RepeatSale_Cnt> <Cnt>21</Cnt> </Table>
</OpenBatchSummary></string>
```

## ***IsCommercialCard***

This Web service operation checks whether the card number entered is for a commercial card or not. The URL to access this Web service is:  
`https://localhost/SmartPayments/validate.asmx?op=IsCommercialCard`. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Please note due to ever-changing card bin-ranges and other factors, not all Commercial/Purchase cards can be definitely determined by this method. Developers should design their applications while keeping this fact in mind. Descriptions of the parameters are listed below.

| Parameter         | Description                           |
|-------------------|---------------------------------------|
| <b>CardNumber</b> | Required. The number of a credit card |

### **Examples**

The following tables contain sample data you can use to test this Web service. The return result will be either true or false.

**Example 1:** The example data shown will result in return false.

| Parameter         | Value            |
|-------------------|------------------|
| <b>CardNumber</b> | 5454545454545454 |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>  
<boolean xmlns="http://localhost/SmartPayments/">false</boolean>
```

**Example 2:** The example data shown will result in return true.

| Parameter         | Value            |
|-------------------|------------------|
| <b>CardNumber</b> | 4055011111111111 |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>  
<boolean xmlns="http://localhost/SmartPayments/">>true</boolean>
```

## ProcessCheck

This Web service operation processes check transactions for a merchant. The URL to access this Web service is:

<https://localhost/SmartPayments/transact.aspx?op=ProcessCheck>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter          | Value   |
|--------------------|---|
| <b>UserName</b>    | Required. User name assigned in the payment server  |
| <b>Password</b>    | Required. Password for the user name assigned in the payment server   |
| <b>TransType</b>   | Required. Type of the check transaction. Valid values are: <ul style="list-style-type: none"> <li>• <b>Sale</b> to make a purchase with a check</li> <li>• <b>Auth (Verify)</b> to authorize/verify an amount of a check</li> <li>• <b>Return</b> to return the money of a settled check transaction to the check holder</li> <li>• <b>Void</b> to undo an unsettled check transaction</li> <li>• <b>Force</b> to force a previous <b>Sale</b> transaction into the current batch (ForceSale)</li> <li>• <b>Capture</b> to settle a single transaction in the current batch; only for terminal-based processors</li> <li>• <b>CaptureAll</b> to settle all transactions in the current batch; only for terminal-based processors</li> </ul> |
| <b>CheckNum</b>    | Required except for these <b>TransType</b> 's: <b>Void, Capture, CaptureAll</b> . Check number uniquely identifies each individual check  |
| <b>TransitNum</b>  | Required except for these <b>TransType</b> 's: <b>Void, Capture, CaptureAll</b> . Transit number uniquely identifies a bank routing number  |
| <b>AccountNum</b>  | Required except for these <b>TransType</b> 's: <b>Void, Capture, CaptureAll</b> . Account number uniquely identifies a check holder's bank account number   |
| <b>Amount</b>      | Required except for these <b>TransType</b> 's: <b>Void, Capture, CaptureAll</b> . The total transaction amount in DDDD.CC format  |
| <b>MICR</b>        | Optional. The Magnetic Ink Check Reader data line, which includes <b>TransitNum</b> , and <b>AccountNum</b> . Required for processing Check-Present transactions  |
| <b>NameOnCheck</b> | Required except for these <b>TransType</b> 's: <b>Void, Capture, CaptureAll</b> . The check holder's name as it appears on the check. The parameter may be required, depending on the merchant's processor setup. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details  |
| <b>DL</b>          | Optional. The check holder's driver's license number. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details  |
| <b>SS</b>          | Optional. The check holder's Social Security Number. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details   |
| <b>DOB</b>         | Optional. The check holder's date of birth. This parameter will remove invalid  |

|                  |  |
|------------------|--|
|                  | characters. See list of <a href="#">Removed Characters</a> for more details  |
| <b>StateCode</b> | Optional. The check holder's 2 character state code. The parameter may be required depending on the merchant's processor setup. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details   |
| <b>CheckType</b> | Optional. The type of the check. Valid values are: <ul style="list-style-type: none"> <li>• <b>Personal</b></li> <li>• <b>Corporate</b></li> <li>• <b>Government</b></li> </ul>  |
| <b>ExtData</b>   | <p>Extended data in XML format</p> <p>These tags may be required for <b>Sale</b> and <b>Return</b> transactions depending on the merchant's processor setup: <b>CityOfAccount</b>, <b>BillToStreet</b>, and <b>BillToPostalCode</b>.</p> <p>Required tag for <b>Return</b>, <b>Void</b>, <b>Force</b>, and <b>Capture</b> transactions is: <b>PNRef</b>.</p> <p><b>RawMICR</b> tag is required for processing Check-Present transactions.</p> <p>Valid values are:</p> <ul style="list-style-type: none"> <li>• <code>&lt;TimeOut&gt;TimeOut&lt;/TimeOut&gt;</code> for timeout value in seconds (default = 30)</li> <li>• <code>&lt;PNRef&gt;PNRef&lt;/PNRef&gt;</code> for a reference number assigned by the payment server</li> <li>• <code>&lt;Phone&gt;Phone&lt;/Phone&gt;</code> for phone number. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> <li>• <code>&lt;Email&gt;EMail&lt;/EMail&gt;</code> for email address. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> <li>• <code>&lt;RawMICR&gt;RawMICR&lt;/RawMICR&gt;</code> for raw Magnetic Ink Check Reader data from the check reader in the format of:<br/><i>TransitNumTAccountNumOCheckNum</i></li> <li>• <code>&lt;InvNum&gt;InvNum&lt;/InvNum&gt;</code> for invoice tracking number. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> <li>• <code>&lt;TrainingMode&gt;TrainingMode&lt;/TrainingMode&gt;</code> to process transaction in Training Mode; either <b>T</b> or <b>F</b></li> <li>• <code>&lt;AllianceNum&gt;AllianceNum&lt;/AllianceNum&gt;</code> is the Alliance number for the check</li> <li>• <code>&lt;AccountType&gt;AccountType&lt;/AccountType&gt;</code> is the type bank account for the check. Valid values are <b>Checking</b> or <b>Saving</b></li> <li>• <code>&lt;CityOfAccount&gt;CityOfAccount&lt;/CityOfAccount&gt;</code> for city name of the check holder's residential address. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> <li>• <code>&lt;BillToStreet&gt;BillToStreet&lt;/BillToStreet&gt;</code> for street name of the</li> </ul> |

|  |  |
|--|--|
|  | <p>check holder's billing address. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details.</p> <ul style="list-style-type: none"> <li>• <b>&lt;BillToCity&gt;BillToCity&lt;/BillToCity&gt;</b> for city name of the check holder's billing address. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> <li>• <b>&lt;BillToState&gt;BillToState&lt;/BillToState&gt;</b> for the two character state code of the check holder's billing address. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> <li>• <b>&lt;BillToPostalCode&gt;BillToPostalCode&lt;/BillToPostalCode&gt;</b> for zip code of the check holder's billing address. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> <li>• <b>&lt;BillToCountry&gt;BillToCountry&lt;/BillToCountry&gt;</b> for country name of the check holder's billing address. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> <li>• <b>&lt;CustomerID&gt;CustomerID&lt;/CustomerID&gt;</b> for customer ID. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> </ul> |
|--|--|

The following tables contain sample data you can use to test this Web service. The **UserName** and **Password** parameters should be changed when testing the examples yourself.

## Examples

**Example 1:** The example data below will process a manually entered check **Sale** transaction through the payment server.

| Parameter          | Value  |
|--------------------|--|
| <b>UserName</b>    | test   |
| <b>Password</b>    | 123  |
| <b>TransType</b>   | Sale   |
| <b>CheckNum</b>    | 1001   |
| <b>TransitNum</b>  | 123456780  |
| <b>AccountNum</b>  | 1234567890   |
| <b>Amount</b>      | 100.00   |
| <b>NameOnCheck</b> | John Doe   |
| <b>StateCode</b>   | WA   |
| <b>ExtData</b>     | <CityOfAccount>City</CityOfAccount> <BillToStreet> 123</BillToStreet> <BillToPostalCode>99999</BillToPostalCode> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><Message>APPROVAL</
Message><AuthCode>GUEIJQ</AuthCode><PNRef>2400</PNRef>
```

```
</Response>
```

**Example 2:** The example data below will process a swiped check **Sale** transaction through the payment server.

| Parameter          | Value  |
|--------------------|--|
| <b>UserName</b>    | test   |
| <b>Password</b>    | 123  |
| <b>TransType</b>   | Sale   |
| <b>CheckNum</b>    | 1001   |
| <b>TransitNum</b>  | 123456780                                    |
| <b>AccountNum</b>  | 1234567890                                   |
| <b>Amount</b>      | 100.00                                       |
| <b>MICR</b>        | 1234567801234567890                          |
| <b>NameOnCheck</b> | John Doe                                     |
| <b>StateCode</b>   | WA   |
| <b>ExtData</b>     | <RawMICR>123456780T1234567890O1001</RawMICR> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><Message>APPROVAL</Message><AuthCode>AUTH NUM 121-
704</AuthCode><PNRef>11622</PNRef>
```

```
</Response>
```

**Example 3:** The example data below will process a manually-entered check **Return** transaction through the payment server. The **PNRef** element in the **ExtData** parameter should be changed when testing this example yourself.

| Parameter          | Value  |
|--------------------|--|
| <b>UserName</b>    | test   |
| <b>Password</b>    | 123  |
| <b>TransType</b>   | Return   |
| <b>CheckNum</b>    | 100  |
| <b>TransitNum</b>  | 123456780  |
| <b>AccountNum</b>  | 1234567890   |
| <b>Amount</b>      | 100.00   |
| <b>NameOnCheck</b> | John Doe   |
| <b>StateCode</b>   | WA   |
| <b>ExtData</b>     | <PNRef>821</PNRef> <CityOfAccount>Any<br>Town</CityOfAccount> <BillToStreet> 123</BillToStreet> <BillToPostalCode><br>99999</BillToPostalCode> <InvNum>1001</InvNum> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><Message>APPROVAL</Message>
<AuthCode>GWNICP</AuthCode><PNRef>2406</PNRef><ExtData>InvNum=100
1</ExtData>
```

```
</Response>
```

**Example 4:** The example data below will process a check **Void** transaction through the payment server. The **PNRef** element in the **ExtData** parameter should be changed when testing this example yourself.

| Parameter        | Value               |
|------------------|---------------------|
| <b>UserName</b>  | test                |
| <b>Password</b>  | 123                 |
| <b>TransType</b> | Void                |
| <b>ExtData</b>   | <PNRef>2405</PNRef> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><Message>APPROVAL</
Message><AuthCode>GW5NPQ</AuthCode><PNRef>2405</PNRef>
```

```
</Response>
```

## ***ProcessCreditCard***

This Web service operation processes credit card transactions for a merchant. The URL to access this Web service is:

<https://localhost/SmartPayments/transact.asmx?op=ProcessCreditCard>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter        | Value  |
|------------------|--|
| <b>UserName</b>  | Required. User name assigned in the payment server   |
| <b>Password</b>  | Required. Password for the user name assigned in the payment server  |
| <b>TransType</b> | Required. Type of the credit card transaction. Valid values are: <ul style="list-style-type: none"><li>• <b>Sale</b> to make a purchase on a credit card</li><li>• <b>Adjustment</b> is used to modify an existing tip amount for an original sale. This applies to the processors that support restaurant adjustment transactions</li><li>• <b>Auth</b> to authorize an amount on a credit card</li><li>• <b>Return</b> to credit the cardholder’s account</li><li>• <b>Void</b> to undo an unsettled transaction. Note: pass the Card Number and ExpDate with null values on voids</li><li>• <b>Force</b> to place an <b>Auth</b> transaction into the current batch (PostAuth) or to place a transaction not processed through the payment server into the current batch (ForceAuth)</li><li>• <b>Capture</b> to settle a single transaction in the current batch; only for terminal-based processors</li><li>• <b>CaptureAll</b> to settle all transactions in the current batch; only for terminal-based processors or host-based processors that support a batch release feature</li><li>• <b>RepeatSale</b> to perform a recurring billing or installment payment transaction</li></ul> |
| <b>CardNum</b>   | Optional except for these <b>TransType</b> ’s: <b>Sale, Auth, Return, Force</b> (ForceAuth).   |

|                   |  |
|-------------------|--|
|                   | Credit card number to process the transaction. For all other transaction types the parameter needs to be included  |
| <b>ExpDate</b>    | Optional except for these <b>TransType</b> 's: <b>Sale, Auth, Return, Force</b> (ForceAuth). Credit card's expiration date in MMY format. For all other transaction types the parameter needs to be included   |
| <b>MagData</b>    | Optional except when processing swiped card transactions. Data located on the track 2 of the magnetic strip of the card. Once this field is populated, the transaction will be indicated as <i>Card-Present</i> transaction and usually result in a more favorable retail discount rate. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details<br><br>The format of the MagData (or Track 2 data) is CardNum=ExpDate followed by the service code and checksum. For example,<br>36438999960016=05121015432112345678   |
| <b>NameOnCard</b> | Optional, depending on different merchant processor setups. The cardholder's name as it appears on the card. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details  |
| <b>Amount</b>     | Optional except for these <b>TransType</b> 's: <b>Auth, Sale, Return, Force</b> (both PostAuth and ForceAuth). The total transaction amount in DDDD.CC format  |
| <b>InvNum</b>     | Optional. Invoice tracking number. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details  |
| <b>PNRef</b>      | Optional except for these <b>TransType</b> 's: <b>Void, Force</b> (PostAuth), <b>Capture</b> . Reference number assigned by the payment server   |
| <b>Zip</b>        | Optional depending on different merchant processor setups. Cardholder's billing address zip code used in address verification. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details  |
| <b>Street</b>     | Optional depending on different merchant processor setup. Cardholder's billing street address used in address verification. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details   |
| <b>CVNum</b>      | Optional. Card verification number   |
| <b>ExtData</b>    | Optional except in the cases of: <b>AuthCode</b> (required for a <b>Force</b> (ForceAuth) transaction); <b>City</b> and <b>BillToState</b> (required by certain processors); <b>Invoice</b> and associated nested data elements (required for Concord EFS Purchase Card Level 3 and Fuel purchases- see section below). Extended data in XML format. Valid values are: <ul style="list-style-type: none"> <li>• <b>&lt;AuthCode&gt;</b><i>ApprovalCode</i><b>&lt;/AuthCode&gt;</b> for original authorization code</li> <li>• <b>&lt;CustCode&gt;</b><i>CustomerCode</i><b>&lt;/CustCode&gt;</b> for customer code or PO number of the customer. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details (Note* This is the tag to be passed for Level II information for Global Payments only, please use PONUM for the other processors that support level II.)</li> <li>• <b>&lt;TipAmt&gt;</b><i>TipAmount</i><b>&lt;/TipAmt&gt;</b> for tip amount in DDDD.CC format</li> <li>• <b>&lt;TaxAmt&gt;</b><i>TaxAmount</i><b>&lt;/TaxAmt&gt;</b> for tax amount in DDDD.CC format</li> <li>• <b>&lt;SequenceNum&gt;</b><i>SequenceNum</i><b>&lt;/SequenceNum&gt;</b> for sequence number used with RepeatSale installment transactions; this designates which number in the sequence the transaction is; it must be a positive integer smaller than or equal to the SequenceCount</li> <li>• <b>&lt;SequenceCount&gt;</b><i>SequenceCount</i><b>&lt;/SequenceCount&gt;</b> for sequence count used with RepeatSale installment transactions; this designates the total</li> </ul> |

number of charges that will be made; it must be a positive integer greater than or equal to the SequenceNum

- **<ServerID>***ServerID***</ServerID>** for a unique server identification number. The data within this XML tag parameter will remove invalid characters. See list of [Removed Characters](#) for more details
- **<TimeOut>***TimeOut***</TimeOut>** for timeout value in seconds (default = 40)
- **<TrainingMode>***TrainingMode***</TrainingMode>** to process transaction in Training Mode; either **T** or **F**
- **<Force>***Force***</Force>** for forcing duplicate transactions to be processed; either **T** or **F**. Note that some processors, including Concord EFS, will not utilize this tag and may still reject a duplicate transaction
- **<RegisterNum>***RegisterNum***</RegisterNum>** for register number. The data within this XML tag parameter will remove invalid characters. See list of [Removed Characters](#) for more details
- **<City>***City***</City>** for the city name of the cardholder's billing address. The data within this XML tag parameter will remove invalid characters. See list of [Removed Characters](#) for more details
- **<BillToState>***State***</BillToState>** for the two character state code of the cardholder's billing address. The data within this XML tag parameter will remove invalid characters. See list of [Removed Characters](#) for more details
- **<CustomerID>***CustomerID***</CustomerID>** for customer ID
- **<PONum>***PONum***</PONum>** for purchase order number. The data within this XML tag parameter will remove invalid characters. See list of [Removed Characters](#) for more details (Note\* This is to be used for Level II information except Global Payments)
- **<BillPayment>***BillPayment***</BillPayment>** to indicate that a transaction is a utility bill payment; either **T** or **F**; only supported for **TransType's Sale** and **RepeatSale**; This tag is only relevant to Retail, MOTO, and eCommerce markets. Currently, this information is only supported for Vital, First Data North, and Global Payments processors; other processors may be supported in the future

• **<Authentication>**

**<XID>***AuthenticationID***</XID>** **Verified By VISA**

**<CAVV>***CAVV***</CAVV>** **CAVV response value**

**<UCAF>***UCAF***</UCAF>** **Universal Card Holder Authentication Field**

Verified by Visa and Universal Cardholder Authentication Field are programs implemented by Visa and MasterCard respectively to verify that an account number is being submitted by the cardholder. These programs are for the Ecommerce market exclusively and only MasterCard and Visa cards support this feature. The data for Visa and MasterCard is different. There is a possible CAVV response for Visa cards that will be returned via ExtData \* Please see sample in the section Examples for Verified by VISA and UCAF for Mastercard.

**<TrustAttempt>***T***</TrustAttempt>**

The trust attempt tag will indicate VBV or MS was attempted but no XID, CAVV, or UCAFF is available for the transaction.

**</Authentication>**

- **<CVPresence>***CVPresence Value***</CVPresence>** CVV2 / CVC2 / CID Presence, indicates whether a CVV2 or CID has been sent along with the request. The valid values for this tag are: **None, NotSubmitted, Submitted Illegible, NotPresent (Not present on card)**. \*Please see sample request and response for this CVPresence tag field.
- **<EntryMode>***EntryModeValue***</EntryMode>** this was added to indicate how the values for the payment information were obtained. The Valid values for this tag are: UNKNOWN, MANUAL, MAGNETICSTRIPE, ICC, and PROXIMITY
- **<Invoice>***Invoice***</Invoice>** to indicate invoice details will be included. Required for Concord EFS Purchase Card Level 3 but optional for fuel purchases. However, fuel purchases must contain the **<Items>** element for transactions of type **Sale** and **Force**. See below for hierarchy of required and optional elements nested within. Please note that all elements included inside **<Invoice>** must be in the specific order listed below

❖ **<Invoice>**

- **<InvNum>***InvNum***</InvNum>** Purchase invoice number
- **<Date>***Date***</Date>** Date of invoice in YYYYMMDD format for Concord
- **<BillTo>**
  - **<CustomerId>***CustomerId***</CustomerId>** Customer ID number
  - **<Name>***Name***</Name>** Customer name
  - **<Address>** Customer address
    - **<Street>***Street***</Street>** Customer address street
    - **<City>***City***</City>** Customer address city
    - **<State>***State***</State>** Customer address state
    - **<Zip>***Zip***</Zip>** Customer address zip code
    - **<Country>***Country***</Country>** Customer address country
  - **</Address>**
  - **<Email>***Email***</Email>** Customer email
  - **<Phone>***Phone***</Phone>** Customer phone number
  - **<Fax>***Fax***</Fax>** Customer fax number
  - **<CustCode>***CustCode***</CustCode>** Customer code
  - **<PONum>***PONum***</PONum>** Purchase order number from customer
  - **<TaxExempt>***TaxExempt***</TaxExempt>** Customer tax exempt status
- **</BillTo>**
- **<Description>***Description***</Description>** Description of purchase
- **<Items>** Required for Concord EFS Purchase Card Level 3 and fuel purchases. Items contained in invoice. Contains one or more **<Item>**

elements

- **<Item>** Required for Concord EFS Purchase Card Level 3 and fuel card purchases. One item in invoice (item details nested within). There may be multiple **<Item>** nested within **<Items>** tag
  - **<SKU>SKU</SKU>** SKU number of item
  - **<UPC>UPC</UPC>** Required for Concord EFS Purchase Card Level 3 and fuel purchases. UPC number of item for Purchase Card Level 3 or the NACS (National Association of Convenience Stores) product code for fuel purchases. The NACS is an industry standard list that Concord is utilizing. For a list of NACS product codes, please contact EFSnet™ customer support at [support@concordefsn.com](mailto:support@concordefsn.com) or 1-877-852-2637.
  - **<Description>Description</Description>** Item description
  - **<Quantity>Quantity</Quantity>** Required for Concord EFS Purchase Card Level 3 and fuel purchases. Quantity purchased of item
  - **<UnitOfMeasurement>UnitOfMeasure</UnitOfMeasurement>** Unit of measurement for item
  - **<UnitPrice>UnitPrice</UnitPrice>** Required for Concord EFS Purchase Card Level 3 and fuel purchases. Unit price of item
  - **<DiscountAmt>DiscountAmount</DiscountAmt>** Possible discount amount on item
  - **<TaxAmt>TaxAmt</TaxAmt>** Possible tax amount on item
  - **<TotalAmt>TotalAmt</TotalAmt>** Total amount of item
  - **<Category>Category</Category>** Required for Concord EFS Purchase Card Level 3 and fuel purchases. Item category for Purchase Card Level 3 or the specific value **Fuel** to designate a fuel purchase item
  - **<TaxRate>TaxRate</TaxRate>** Possible tax rate applied to item
- **</Item>**
- **</Items>**
- **<DiscountAmt>DiscountAmount</DiscountAmt>** Possible total discount for invoice
- **<ShippingAmt>ShippingAmt</ShippingAmt>** Possible shipping amount for invoice
- **<DutyAmt>DutyAmt</DutyAmt>** Possible duty amount for invoice
- **<TaxAmt>TaxAmt</TaxAmt>** Possible tax amount for invoice
- **<NationalTaxInc>NationalTaxInc</NationalTaxInc>** Possible additional tax amount included in invoice total
- **<TotalAmt>TotalAmt</TotalAmt>** Total amount of the transaction on the invoice
- ❖ **</Invoice>**
  - **<Fleet>Fleet</Fleet>** Required for Concord EFS Fleet card purchases. Information on fleet member making purchase. See below for hierarchy of elements nested within. Please note that all elements included inside **<Fleet>**

|  |   |
|--|---|
|  | <p>must be in the specific order listed below</p> <ul style="list-style-type: none"> <li>❖ <b>&lt;Fleet&gt;</b> <ul style="list-style-type: none"> <li>➤ <b>&lt;VehicleNum&gt;</b><i>VehicleNum</i><b>&lt;/VehicleNum&gt;</b> May be required for specific Concord EFS Fleet card purchases. The vehicle number</li> <li>➤ <b>&lt;DriverNum&gt;</b><i>DriverNum</i><b>&lt;/DriverNum&gt;</b> May be required for specific Concord EFS Fleet card purchases. The vehicle driver's number</li> <li>➤ <b>&lt;OdometerReading&gt;</b><i>OdometerReading</i><b>&lt;/OdometerReading&gt;</b> May be required for specific Concord EFS Fleet card purchases. The current odometer reading of the fleet vehicle</li> </ul> </li> <li>❖ <b>&lt;/Fleet&gt;</b></li> <li>❖ <b>&lt;CardType&gt;</b><i>CardType</i><b>&lt;/CardType&gt;</b> <ul style="list-style-type: none"> <li>➤ When <b>TransType</b> does not equal <b>Capture</b> or <b>CaptureAll</b>: Required for manually-entered fleet card transactions that have the ISO prefix of the fleet card present only in the track data and not in the embossed data on the front of the card. Valid values are <b>WEX</b> and <b>Voyager</b>. Concord currently does not support manually entered Voyager cards</li> <li>➤ When <b>TransType</b> equals <b>CaptureAll</b>: Optional valid values can be: <b>ALL</b> to specify all payment methods assigned to the merchant account should be settled, or a combination of the specific payment methods separated by a colon (i.e. <b>CREDIT:DEBIT:EBT:EGC</b>) in order to specify which individual payment methods assigned to the merchant account should be settled. Please note that if the processor requires all payment methods to settle at the same time, it is required to use the <b>ALL</b> value or the appropriate combination of the specific payment methods in order to settle the account correctly. Currently, only host-based processors that support a manual batch settlement (or batch release) require all payment methods to be settled at the same time</li> <li>➤ When <b>TransType</b> equals <b>Capture</b>: This element does not apply since only 1 transaction will be settled</li> </ul> </li> </ul> |
|  |   |

### Purchase Card Level 3 Data Use

Level 3 data on Purchase Card transactions is now available, but currently only through the Concord EFS processor. This sends invoice information with line-item detail provided to the processor for validation. The data is all sent through the **ExtData** parameter nested within the **<Invoice>** element (see chart above). Note that if you use the **<Invoice>** element it will ignore any data of the same purpose specified elsewhere by way of a parameter or other **ExtData** parameter tag, i.e. it will use one or the other but not both. For example, if you supply the **<TaxAmt>** element within the **<Invoice>** element *and* as a separate element in the **ExtData** parameter, the separate **<TaxAmt>** element outside the **<Invoice>** element will be ignored (duplicate information will not cause a problem). See Example 10 below for a sample transaction sending Purchase Card Level 3 data.

## Fuel Purchases: Standard and Fleet Card Use

Credit card processing for fuel purchases with both Standard and Fleet type cards are now available, currently through Concord EFS only. This functionality allows for fuel purchases with standard credit cards (Visa, Mastercard, etc) and with Fleet type cards (Wex, Voyager, and MasterCard Fleet are currently supported). Fuel purchases are differentiated at the gateway from other purchases by the **Fuel** designation placed within the **<Category>** tag in item descriptions (see Examples 11 through 13 below). In effect, a transaction will only be treated as a fuel transaction if at least one of the items within **<Items>** is designated as category **Fuel**.

Both Standard and Fleet cards require item-level purchase information for fuel purchases (for **TransType**'s **Sale** and **Force**), and Fleet cards may additionally require vehicle number, driver number, and/or odometer information on such purchases. If all the required information for a certain purchase is not provided, the transaction will be rejected and an error message generated. Note that Fleet cards in some cases can be used to purchase non-fuel items on a transaction designated as fuel, but item-level information must be present for all items in the transaction, otherwise the transaction may be declined. The main implication for the developer is that additional data must be passed to the gateway in order for fuel purchases to process correctly.

For standard credit card processing on **Sale** and **Force** fuel transactions, item-level purchase information must be provided. It can be passed inside the **<Items>** tag alone or nestled within **<Invoice>** information in **ExtData**. See above chart for details on these and other required XML tags for standard card processing on fuel transactions, and see Example 11 below.

For Fleet card processing on fuel purchases, additional information on the fleet member such as vehicle number, driver number, and/or odometer information may also need to be provided (according to the requirements for the particular Fleet card; for example, Wex typically requires the **<DriverNum>** tag). See the **<Fleet>** tag as described in the table above, and Examples 12 and 13 below. Fleet data must be provided on **Sale**, **Auth**, **Force**, and **Return** transactions. This information will be saved and, if not obtained a second time for a **Force** (PostAuth), will be automatically sent to the processor.

The Fleet data can generally come directly from the card's magnetic data or the POS system can acquire the data by examining information found on the card's magnetic data and then prompting the cardholder for the required data. When transactions are submitted, the card's requirements will be validated by the payment processor and an error will be generated if the proper Fleet data is not submitted. Exceptions: If a MasterCard Fleet card is used and Fleet information is not provided, the transaction will be processed as a standard fuel transaction, rather than generating an error. Also, MasterCard Fleet allows the user to exclude fleet data on manually-entered transactions.

Manual Fleet transactions present a special case. Fleet cards are different in that the account data embossed on a card is often not the same as the track (magnetic) data. This

is true for both Wex and Voyager Fleet cards. The result of this is that in a manually-entered (non-swiped) card submission, the card type cannot be identified by the account number data displayed on the card. For a Wex card, the user must identify the card manually, and this information must be passed in **<CardType>** tag in the **ExtData** parameter (see Example 13). However, a Voyager card cannot be processed manually through Concord EFS at this time.

## Examples

The following tables contain sample data you can use to test this Web service. The **UserName** and **Password** parameters should be changed when testing the examples yourself.

**Example 1:** The example data below will process a manually-entered credit card **Sale** transaction through the payment server. It will be processed even if it is a duplicate transaction.

| Parameter         | Value            |
|-------------------|------------------|
| <b>UserName</b>   | test             |
| <b>Password</b>   | 123              |
| <b>TransType</b>  | Sale             |
| <b>CardNum</b>    | 5454545454545454 |
| <b>ExpDate</b>    | 0509             |
| <b>NameOnCard</b> | John Doe         |
| <b>Amount</b>     | 1.00             |
| <b>ExtData</b>    | <Force>T</Force> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><Message>APPROVAL</
  Message><AuthCode>006063</AuthCode><PNRef>2296</PNRef><HostCode>0
  3530EVV3K2ZA2F453Q</HostCode><GetAVSResult>N</GetAVSResult><GetAVS
  ResultTXT>No Match</GetAVSResultTXT><GetStreetMatchTXT>No
  Match</GetStreetMatchTXT><GetZipMatchTXT>No
  Match</GetZipMatchTXT><GetCommercialCard>False</GetCommercialCard><Ext
  Data>CardType=MASTERCARD</ExtData>
```

```
</Response>
```

**Example 2:** The example data below will process a manually-entered credit card **Auth** transaction as a commercial card through the payment server.

| Parameter         | Value  |
|-------------------|--|
| <b>UserName</b>   | test   |
| <b>Password</b>   | 123  |
| <b>TransType</b>  | Auth   |
| <b>CardNum</b>    | 4055016727870315                               |
| <b>ExpDate</b>    | 0509   |
| <b>NameOnCard</b> | John Doe                                       |
| <b>Amount</b>     | 1.00   |
| <b>InvNum</b>     | 1001   |
| <b>ExtData</b>    | <TaxAmt>0.50</TaxAmt> <CustCode>102</CustCode> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><Message>APPROVAL
  VITAL1</Message><AuthCode>VITAL1</AuthCode><PNRef>2275</PNRef><GetComm
  ercialCard>True</GetCommercialCard><ExtData>InvNum=1001,CardType=VISA
  </ExtData>
```

```
</Response>
```

**Example 3:** The example data below will process a swiped credit card **Return** transaction through the payment server.

| Parameter         | Value                                |
|-------------------|--------------------------------------|
| <b>UserName</b>   | test                                 |
| <b>Password</b>   | 123                                  |
| <b>TransType</b>  | Return                               |
| <b>CardNum</b>    | 371449635398431                      |
| <b>ExpDate</b>    | 1205                                 |
| <b>MagData</b>    | 371449635398431=05121015432112345678 |
| <b>NameOnCard</b> | John Doe                             |

|               |      |
|---------------|------|
| <b>Amount</b> | 1.00 |
|---------------|------|

Result:

```
<?xml version="1.0" encoding="utf-8" ?>

<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">

<Result>0</Result><RespMSG>Approved</RespMSG><Message>APPROVAL
  VITAL7</Message><AuthCode>VITAL7</AuthCode><PNRef>2307</PNRef><GetComm
  ercialCard>False</GetCommercialCard><ExtData>CardType=AMEX</ExtData>

</Response>
```

**Example 4:** The example data below will process a credit card **Void** transaction through the payment server. The **PNRef** parameter should be changed when testing this example yourself. **Note:** pass the card number and ExpDate with null values on voids

| Parameter        | Value |
|------------------|-------|
| <b>UserName</b>  | test  |
| <b>Password</b>  | 123   |
| <b>TransType</b> | Void  |
| <b>PNRef</b>     | 2308  |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>

<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">

<Result>0</Result><RespMSG>Approved</RespMSG><AuthCode>VITAL4</AuthCode>
  <PNRef>2309</PNRef>

</Response>
```

**Example 5:** The example data below will process a credit card **Force** (PostAuth) transaction through the payment server using Training Mode. The **PNRef** parameter should be changed when testing this example yourself.

| Parameter         | Value                          |
|-------------------|--------------------------------|
| <b>UserName</b>   | test                           |
| <b>Password</b>   | 123                            |
| <b>TransType</b>  | Force                          |
| <b>NameOnCard</b> | John Doe                       |
| <b>Amount</b>     | 1.00                           |
| <b>PNRef</b>      | 2310                           |
| <b>ExtData</b>    | <TrainingMode>T</TrainingMode> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
<Result>0</Result><RespMSG>Approved</RespMSG><AuthCode>DEMO-
  2</AuthCode><PNRef>2318</PNRef>
</Response>
```

**Example 6:** The example data below will process a manually-entered credit card **Force** (ForceAuth) transaction through the payment server.

| Parameter         | Value                       |
|-------------------|-----------------------------|
| <b>UserName</b>   | test                        |
| <b>Password</b>   | 123                         |
| <b>TransType</b>  | Force                       |
| <b>CardNum</b>    | 5454545454545454            |
| <b>ExpData</b>    | 0509                        |
| <b>NameOnCard</b> | John Doe                    |
| <b>Amount</b>     | 1.00                        |
| <b>ExtData</b>    | <AuthCode>123456</AuthCode> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```

<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">

  <Result>0</Result><RespMSG>Approved</RespMSG><Message>APPROVAL</Message><AuthCode>123456</AuthCode><PNRef>2317</PNRef><GetCommercialCard>False
  </GetCommercialCard><ExtData>CardType=MASTERCARD</ExtData>

</Response>

```

**Example 7:** The example data below will process a credit card **Capture** transaction through the payment server to settle a single transaction in the current batch. The **PNRef** parameter should be changed when testing this example yourself.

| Parameter        | Value   |
|------------------|---------|
| <b>UserName</b>  | test    |
| <b>Password</b>  | 123     |
| <b>TransType</b> | Capture |
| <b>PNRef</b>     | 2327    |

Result:

```

<?xml version="1.0" encoding="utf-8" ?>

<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">

  <Result>0</Result><RespMSG>Approved</RespMSG><Message>ACCEPTED</Message><AuthCode>GB00029
  ACCEPTED</AuthCode><ExtData>Net_Count=1,Net_Amount=1,Settle_DT=20
  04-04-13 15:36:26</ExtData>

</Response>

```

**Example 8:** The example data below will process a credit card **CaptureAll** transaction through the payment server to settle all transactions in the current batch.

| Parameter        | Value      |
|------------------|------------|
| <b>UserName</b>  | test       |
| <b>Password</b>  | 123        |
| <b>TransType</b> | CaptureAll |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><Message>ACCEPTED</M
essage><AuthCode>GB0030
ACCEPTED</AuthCode><ExtData>Net_Count=1,Net_Amount=1,Settle_DT=20
04-04-13 15:42:45</ExtData>
```

```
</Response>
```

## Batch Close Response Detail Detail

There is a programmatic way to add batch detail in the responses for the settlement commands. To avail of this feature, there would be an adjustment to the SQL database and would need to add the following key values in the AppSetting\_T

8, ExpandExtDataWhen, ALWAYS NEVER EXTERNAL or INTERNAL, Null

Always = Always send the information to all interfaces.  
Never = Never send the information to any interface.  
External = Send the information to PAY and Web Services only.  
Internal (Default) = Send the information to the Virtual Terminal

The integrator or Database Administrator may avail of different methods to add these values but this is a simple SQL script example on how to do this is:

Use PayServerSQLV2k5

```
Insert into AppSetting_T (Application_Key, AppSetting_Key, AppSetting_Value, XmlProfile_TXT)
VALUES ('8', 'ExpandExtDataWhen', 'ALWAYS', NULL)
```

## Examples of Responses with Single/Multiple Batch Detail Enabled

The following examples highlight the additions that are available for multi-batch settlements.

### 1. Single Batch Success

Response (Web service)

```
<Response><Result>0</Result><RespMSG>Approved</RespMSG><Message>ACCEP
TED</Message><Message1></Message1><Message2></Message2><AuthCode>GB00
```

```
258 ACCEPTED</AuthCode><PNRef></PNRef><HostCode>GB00258
ACCEPTED</HostCode><HostURL></HostURL><ReceiptURL></ReceiptURL><Get
AVSResult></GetAVSResult><GetAVSResultTXT></GetAVSResultTXT><GetStreet
MatchTXT></GetStreetMatchTXT><GetZipMatchTXT></GetZipMatchTXT><GetCV
Result></GetCVResult><GetCVResultTXT></GetCVResultTXT><GetGetOrigResult>
</GetGetOrigResult><GetCommercialCard>False</GetCommercialCard><WorkingKey
></WorkingKey><KeyPointer></KeyPointer><ExtData>CardType=ALL,Net_Count=3,
Net_Amount=-3.00,Settle_DT=2006-06-27
12:41:52,BatchNum=258,Batch=<Summary>Net_Count=3,Net_Amount=-
3.00,Settle_DT=2006-06-27
12:41:52,Result=0</Summary><Detail>Net_Count=3,Net_Amount=-
3.00,Settle_DT=2006-06-27 12:41:52,Result=0,Number=258,AuthCode=GB00258
ACCEPTED,Message=ACCEPTED</Detail></ExtData></Response>
```

## 2. Single Batch Failure

### Response (Web service)

```
<Response><Result>12</Result><RespMSG>Decline</RespMSG><Message>RB E
0004 D
24</Message><Message1></Message1><Message2></Message2><AuthCode></AuthC
ode><PNRef></PNRef><HostCode></HostCode><HostURL></HostURL><ReceiptUR
L></ReceiptURL><GetAVSResult></GetAVSResult><GetAVSResultTXT></GetAVS
ResultTXT><GetStreetMatchTXT></GetStreetMatchTXT><GetZipMatchTXT></GetZi
pMatchTXT><GetCVResult></GetCVResult><GetCVResultTXT></GetCVResultTXT
><GetGetOrigResult></GetGetOrigResult><GetCommercialCard>False</GetCommerci
alCard><WorkingKey></WorkingKey><KeyPointer></KeyPointer><ExtData>CardTyp
e=ALL,Batch=<Summary>Net_Count=0,Net_Amount=.00,Settle_DT=2006-06-27
12:46:07,Result=12</Summary><Detail>Net_Count=3,Net_Amount=-
3.00,Settle_DT=2006-06-27 12:46:07,Result=12,Number=262,Message=RB E 0004 D
24</Detail></ExtData></Response>
```

## 3. Mutli Batch Success

### Response (Web service)

```
<Response><Result>0</Result><RespMSG>Approved</RespMSG><Message>ACCEP
TED</Message><Message1></Message1><Message2></Message2><AuthCode>GB00
261 ACCEPTED</AuthCode><PNRef></PNRef><HostCode>GB00261
ACCEPTED</HostCode><HostURL></HostURL><ReceiptURL></ReceiptURL><Get
AVSResult></GetAVSResult><GetAVSResultTXT></GetAVSResultTXT><GetStreet
MatchTXT></GetStreetMatchTXT><GetZipMatchTXT></GetZipMatchTXT><GetCV
Result></GetCVResult><GetCVResultTXT></GetCVResultTXT><GetGetOrigResult>
</GetGetOrigResult><GetCommercialCard>False</GetCommercialCard><WorkingKey
></WorkingKey><KeyPointer></KeyPointer><ExtData>CardType=ALL,Net_Count=9,
Net_Amount=-9.00,Settle_DT=2006-06-27
```

```

12:43:49,BatchNum=261,Batch=<Summary>Net_Count=9,Net_Amount=-
9.00,Settle_DT=2006-06-27
12:43:49,Result=0</Summary><Detail>Net_Count=4,Net_Amount=-
4.00,Settle_DT=2006-06-27 12:43:49,Result=0,Number=259,AuthCode=GB00259
ACCEPTED,Message=ACCEPTED</Detail><Detail>Net_Count=4,Net_Amount=-
4.00,Settle_DT=2006-06-27 12:43:52,Result=0,Number=260,AuthCode=GB00260
ACCEPTED,Message=ACCEPTED</Detail><Detail>Net_Count=1,Net_Amount=-
1.00,Settle_DT=2006-06-27 12:43:54,Result=0,Number=261,AuthCode=GB00261
ACCEPTED,Message=ACCEPTED</Detail></ExtData></Response>

```

#### 4. Multi-Batch Partial Success

##### Response (Web service)

```

<Response><Result>15</Result><RespMSG>Partial</RespMSG><Message>ACCEPT
ED</Message><Message1></Message1><Message2></Message2><AuthCode>GB0026
2 ACCEPTED</AuthCode><PNRef></PNRef><HostCode>GB00262
ACCEPTED</HostCode><HostURL></HostURL><ReceiptURL></ReceiptURL><Get
AVSResult></GetAVSResult><GetAVSResultTXT></GetAVSResultTXT><GetStreet
MatchTXT></GetStreetMatchTXT><GetZipMatchTXT></GetZipMatchTXT><GetCV
Result></GetCVResult><GetCVResultTXT></GetCVResultTXT><GetGetOrigResult>
</GetGetOrigResult><GetCommercialCard>False</GetCommercialCard><WorkingKey
></WorkingKey><KeyPointer></KeyPointer><ExtData>CardType=ALL,Net_Count=1,
Net_Amount=-1.00,Settle_DT=2006-06-27
12:48:15,BatchNum=262,Batch=<Summary>Net_Count=1,Net_Amount=-
1.00,Settle_DT=2006-06-27
12:48:15,Result=15</Summary><Detail>Net_Count=4,Net_Amount=-
4.00,Settle_DT=2006-06-27 12:48:15,Result=12,Number=262,Message=RB E 0004 D
24</Detail><Detail>Net_Count=4,Net_Amount=-4.00,Settle_DT=2006-06-27
12:48:17,Result=12,Number=262,Message=RB E 0006 D
24</Detail><Detail>Net_Count=1,Net_Amount=-1.00,Settle_DT=2006-06-27
12:48:19,Result=0,Number=262,AuthCode=GB00262
ACCEPTED,Message=ACCEPTED</Detail></ExtData></Response>

```

**Example 9:** The example data below will process a credit card **RepeatSale** transaction as a recurring payment through the payment server based on the **PNRef** number of a previous **Sale** transaction. The **PNRef** parameter should be changed when testing this example yourself.

| Parameter        | Value      |
|------------------|------------|
| <b>UserName</b>  | test       |
| <b>Password</b>  | 123        |
| <b>TransType</b> | RepeatSale |
| <b>PNRef</b>     | 2329       |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>

<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">

<Result>0</Result><RespMSG>Approved</RespMSG><Message>APPROVAL
  VITAL2</Message><AuthCode>VITAL2</AuthCode><PNRef>2332</PNRef><GetComm
  ercialCard>False</GetCommercialCard>

</Response>
```

**Example 10:** The example data below will process a credit card **Sale** transaction through the Payment Server, passing the appropriate **ExtData** to provide Purchase Card Level 3 information to the processor. Note that Concord EFS is the only processor which will actually use such data at this time; if such data were sent via **ExtData** to another processor, the transaction would process, but the Level 3 data would not be utilized.

| Parameter         | Value  |
|-------------------|--|
| <b>UserName</b>   | concord  |
| <b>Password</b>   | 123  |
| <b>TransType</b>  | Sale   |
| <b>CardNum</b>    | 5233272716340016   |
| <b>ExpDate</b>    | 0208   |
| <b>MagData</b>    | 5233272716340016=080212121228  |
| <b>NameOnCard</b> | John Doe   |
| <b>Amount</b>     | 26.50  |
| <b>ExtData</b>    | <Invoice><InvNum>123</InvNum><Date>050421</Date><BillTo><CustomerId>CID101</CustomerId><Name>John Doe</Name><Address><Street>123 Main Street</Street><City>Any City</City><State>WA</State><Zip>98052</Zip><Country>USA</Country></Address><Email>test@test.com</Email><Phone>132-123-1234</Phone><Fax>123-123-1235</Fax><CustCode>CCode123</CustCode><PONum>PO123</PONum><TaxExempt>True</TaxExempt></BillTo><Description>One big sale</Description><Items><Item><TotalAmt>1</TotalAmt><UPC>001</UPC><Quantity>1</Quantity><UnitPrice>1</UnitPrice><Category>Fuel</Category><Description>UnLeaded</Description><UnitOfMeasurement>Gallon</UnitOfMeasurement></Item></Items><DiscountAmt>0</DiscountAmt><ShippingAmt>1.11</ShippingAmt><DutyAmt>0</DutyAmt><TaxAmt>1.22</TaxAmt><NationalTaxInc>0</NationalTaxInc><TotalAmt>26.50</TotalAmt></Invoice> |

|  |  |
|--|--|
|  |  |
|--|--|

Result:

```
<?xml version="1.0" encoding="utf-8" ?>

<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">

  <Result>0</Result><RespMSG>OK</RespMSG><Message>APPROVAL</Message
  ><AuthCode>990477</AuthCode><PNRef>38473</PNRef><HostCode>10008
  693148</HostCode><GetCommercialCard>True</GetCommercialCard><ExtData>
  CardType=MASTERCARD</ExtData>

</Response>
```

**Example 11:** The example data below will process a swiped credit card (Mastercard) **Sale** transaction as a fuel transaction through the payment server. Note that **ExtData** contains both required and optional tags for this transaction (see table above).

| Parameter         | Value   |
|-------------------|---|
| <b>UserName</b>   | test  |
| <b>Password</b>   | 123   |
| <b>TransType</b>  | Sale  |
| <b>CardNum</b>    | 5233272716340016  |
| <b>ExpDate</b>    | 0208  |
| <b>MagData</b>    | 5233272716340016=080212121228   |
| <b>NameOnCard</b> | John Doe  |
| <b>Amount</b>     | 26.50   |
| <b>ExtData</b>    | <Items><Item><UPC>001</UPC><Quantity>1</Quantity><UnitPrice>1</UnitPrice><Category>Fuel</Category><Description>UnLeaded</Description><UnitOfMeasurement>Gallon</UnitOfMeasurement></Item></Items> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>OK</RespMSG><Message>APPROVAL</Message
><AuthCode>333333</AuthCode><PNRef>39081</PNRef><HostCode>10008
778024</HostCode><GetCommercialCard>False</GetCommercialCard><ExtData
>CardType=MASTERCARD</ExtData>
```

```
</Response>
```

**Example 12:** The example data below will process a swiped Fleet card **Sale** transaction as a fuel transaction through the payment server. Note that **ExtData** contains both required and optional tags for this transaction (see table above).

| Parameter         | Value  |
|-------------------|--|
| <b>UserName</b>   | test   |
| <b>Password</b>   | 123  |
| <b>TransType</b>  | Sale   |
| <b>CardNum</b>    | 6900460420001234566  |
| <b>ExpDate</b>    | 0306   |
| <b>MagData</b>    | 6900460420001234566=06031000563100000  |
| <b>NameOnCard</b> | John Doe   |
| <b>Amount</b>     | 1.00   |
| <b>ExtData</b>    | <Fleet><DriverNum>123</DriverNum><OdometerReading>78964</OdometerReading></Fleet><Items><Item><UPC>001</UPC><Quantity>1</Quantity><UnitPrice>1</UnitPrice><Category>Fuel</Category><Description>UnLeaded</Description><UnitOfMeasurement>Gallon</UnitOfMeasurement></Item></Items> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>OK</RespMSG><Message>APPROVAL</Message
><AuthCode>333333</AuthCode><PNRef>39082</PNRef><HostCode>10008
778025</HostCode><GetCommercialCard>False</GetCommercialCard><ExtData
>CardType=WEX</ExtData>
```

```
</Response>
```

**Example 13:** The example data below will process a manually-entered Fleet card **Sale** transaction as a fuel transaction through the payment server. Note that the **CardType** must be designated on a manually-entered Fleet transaction. Also note that **ExtData** contains both required and optional tags for this transaction (see table above).

| Parameter         | Value  |
|-------------------|--|
| <b>UserName</b>   | test   |
| <b>Password</b>   | 123  |
| <b>TransType</b>  | Sale   |
| <b>CardNum</b>    | 0420001234566  |
| <b>ExpDate</b>    | 0306   |
| <b>NameOnCard</b> | John Doe   |
| <b>Amount</b>     | 50.00  |
| <b>ExtData</b>    | <Fleet><VehicleNum>321</VehicleNum><DriverNum>123</DriverNum><OdometerReading>78964</OdometerReading></Fleet><Items><Item><UPC>001</UPC><Quantity>1</Quantity><UnitPrice>1</UnitPrice><Category>Fuel</Category><Description>UnLeaded</Description><UnitOfMeasurement>Gallon</UnitOfMeasurement></Item></Items><CardType>WEX</CardType> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
  <Result>0</Result><RespMSG>OK</RespMSG><Message>APPROVAL</Message>
  <AuthCode>333333</AuthCode><PNRef>39083</PNRef><HostCode>10008
  778047</HostCode><GetCommercialCard>False</GetCommercialCard><ExtData
  >CardType=WEX</ExtData>
```

```
</Response>
```

**Example 14.** This example illustrates the new command **ADJUSTMENT** that can modify an original Sale transaction specific to tip adjustment. This is applicable to all restaurant supported processors. This sample starts with an original sale amount with no tip added in the Ext Data field using HTTP POST

| Parameter   | Value   |
|-------------|---|
| UserName:   | <input type="text" value="vitalr"/>           |
| Password:   | <input type="text" value="1234"/>             |
| TransType:  | <input type="text" value="Sale"/>             |
| CardNum:    | <input type="text" value="5439750001500347"/> |
| ExpDate:    | <input type="text" value="1208"/>             |
| MagData:    | <input type="text"/>                          |
| NameOnCard: | <input type="text" value="John Smith"/>       |
| Amount:     | <input type="text" value="12.00"/>            |
| InvNum:     | <input type="text"/>                          |
| PNRef:      | <input type="text"/>                          |
| Zip:        | <input type="text" value="33019"/>            |
| Street:     | <input type="text" value="123 Main Street"/>  |
| CVNum:      | <input type="text"/>                          |
| ExtData:    | <input type="text" value="998"/>              |

**Response :**

<?xml version="1.0" encoding="utf-8" ?>

= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="SmartPayments">

<Result>**0**</Result>

<RespMSG>**Approved**</RespMSG>

<Message>**NO MATCH**</Message>

<AuthCode>**TAS770**</AuthCode>

<PNRef>**27722**</PNRef>

<HostCode>**704623500829**</HostCode>

```
<GetAVSResult>N</GetAVSResult>  
<GetAVSResultTXT>No Match</GetAVSResultTXT>  
<GetStreetMatchTXT>No Match</GetStreetMatchTXT>  
<GetZipMatchTXT>No Match</GetZipMatchTXT>  
<GetCommercialCard>False</GetCommercialCard>  
<ExtData>CardType=MASTERCARD</ExtData>  
  
</Response>
```

*To add a tip to this original Sale transaction, the command **ADJUSTMENT** is used along with the username,login, PNREF of the original sale and tip amount in the extended data field as shown below.*

*Currently only the tip amount is adjustable. Please observe the screenshot below there is no need to resend all the credit card information data to perform a tip adjustment*

| Parameter   | Value  |
|-------------|--|
| UserName:   | <input type="text" value="vitalr"/>                            |
| Password:   | <input type="text" value="1234"/>                              |
| TransType:  | <input type="text" value="Adjustment"/>                        |
| CardNum:    | <input type="text"/>   |
| ExpDate:    | <input type="text"/>   |
| MagData:    | <input type="text"/>   |
| NameOnCard: | <input type="text"/>   |
| Amount:     | <input type="text"/>   |
| InvNum:     | <input type="text"/>   |
| PNRef:      | <input type="text" value="27722"/>                             |
| Zip:        | <input type="text"/>   |
| Street:     | <input type="text"/>   |
| CVNum:      | <input type="text"/>   |
| ExtData:    | <input type="text" value="&lt;TipAmt&gt;3.00&lt;/TipAmt&gt;"/> |

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="SmartPayments">
```

```
<Result>0</Result>
```

```
<RespMSG>Approved</RespMSG>
```

```
<Message>APPROVAL</Message>
```

```
<PNRef>27723</PNRef>
```

```
<GetCommercialCard>False</GetCommercialCard>
```

```
</Response>
```

## Examples for Verified By VISA and UCAF for Mastercard

### VISA Requests

```
<?xml version="1.0" encoding="UTF-8"?><XMLPayRequest version="2"
Timeout="40"><RequestData><Vendor>2</Vendor><Partner>101</Partner><Transacti
ons><Transaction><Sale><PayData><Invoice><BillTo><Address><Street>8320</Stree
t><Zip>85284</Zip></Address></BillTo><ShipTo><Address><Zip>85284</Zip></Ad
dress></ShipTo><TotalAmt>1.00</TotalAmt></Invoice><Tender><Card><CardType>
VISA</CardType><CardNum>4003002345678903</CardNum><ExpDate>0809</ExpD
ate><CVNum>999</CVNum><Amount>1.00</Amount><ExtData>
<Authentication><XID>1234567890123456789012345678901234567890</XID><CAV
V>0987654321098765432109876543210987654321</CAVV>
</Authentication></ExtData></Card></Tender></PayData></Sale></Transaction></Tra
nsactions></RequestData><RequestAuth><UserPass><User>vital</User><Password>12
3</Password></UserPass></RequestAuth></XMLPayRequest>
```

| Parameter   | Value   |
|-------------|---|
| UserName:   | <input type="text" value="vital"/>  |
| Password:   | <input type="text" value="1234"/>   |
| TransType:  | <input type="text" value="Sale"/>   |
| CardNum:    | <input type="text" value="4003002345678903"/>   |
| ExpDate:    | <input type="text" value="0809"/>   |
| MagData:    | <input type="text"/>  |
| NameOnCard: | <input type="text"/>  |
| Amount:     | <input type="text" value="1.00"/>   |
| InvNum:     | <input type="text"/>  |
| PNRef:      | <input type="text"/>  |
| Zip:        | <input type="text"/>  |
| Street:     | <input type="text"/>  |
| CVNum:      | <input type="text" value="999"/>  |
| ExtData:    | <input type="text" value="&lt;Authentication&gt;&lt;XID&gt;1234567890123456789012345"/> |

## Visa Response

```
<?xml version="1.0" encoding="UTF-8"?><XMLPayResponse><ResponseData><Vendor>2</Vendor><Partner>101</Partner><TransactionResults><TransactionResult><Result>0</Result><AVSResult><AVSResponse>Y</AVSResponse><StreetMatch>Y</StreetMatch><ZipMatch>Y</ZipMatch></AVSResult><CVResult>M</CVResult><Message>EXACT MATCH</Message><ExtData><CAVVResponse>2</CAVVResponse></ExtData><PNRef>19219</PNRef><AuthCode>VITAL2</AuthCode><HostCode>514520501676</HostCode><CommercialCard>False</CommercialCard><TransDate>052505</TransDate><TransTime>132956</TransTime></TransactionResult></TransactionResults></ResponseData></XMLPayResponse>
```

## MasterCard Request

```
CC17| ProcessCreditCard| 1| vital| 1234| sale| 5499990123456781 | 0809| | 3.00| | 85284| 2345| 998|<Authentication><UCAF>09876543210987654321098765432109</UCAF></Authentication>
```

| Parameter   | Value   |
|-------------|---|
| UserName:   | <input type="text" value="vital"/>  |
| Password:   | <input type="text" value="1234"/>   |
| TransType:  | <input type="text" value="Sale"/>   |
| CardNum:    | <input type="text" value="5499990123456781"/>   |
| ExpDate:    | <input type="text" value="0809"/>   |
| MagData:    | <input type="text"/>  |
| NameOnCard: | <input type="text" value="John Smith"/>   |
| Amount:     | <input type="text" value="3.00"/>   |
| InvNum:     | <input type="text"/>  |
| PNRef:      | <input type="text"/>  |
| Zip:        | <input type="text" value="85284"/>  |
| Street:     | <input type="text"/>  |
| CVNum:      | <input type="text" value="998"/>  |
| ExtData:    | <input type="text" value="&lt;Authentication&gt;&lt;UCAF&gt;0987654321098765432109876543210987654321098765432109&lt;/UCAF&gt;&lt;/Authentication&gt;"/> |

## Mastercard Response

```
<?xml version="1.0" encoding="UTF-8"?><XMLPayRequest version="2"
Timeout="40"><RequestData><Vendor>2</Vendor><Partner>101</Partner><Transactions><Tra
nsaction><Sale><PayData><Invoice><BillTo><Address><Street>2345</Street><Zip>85284</Zip
></Address></BillTo><ShipTo><Address><Zip>85284</Zip></Address></ShipTo><TotalAmt>3.
00</TotalAmt></Invoice><Tender><Card><CardType>MASTERCARD</CardType><CardNum>
5499990123456781</CardNum><ExpDate>0809</ExpDate><CVNum>998</CVNum><Amount>
3.00</Amount><ExtData><Authentication><UCAF>09876543210987654321098765432109</UC
AF></Authentication><</ExtData></Card></Tender></PayData></Sale></Transaction></Transa
ctions></RequestData><RequestAuth><UserPass><User>vital</User><Password>123</Passwo
rd></UserPass></RequestAuth></XMLPayRequest>
```

## Example of CVPresence

### Request

```
CC61| ProcessCreditCard| 1| fdcnorth| 123| sale| 4012000033330026| 0408| | | 29.95| | |
631461234| 12115 LACKLAND| |<CVPresence>Illegible</CVPresence>
```

### Response

```
<?xml version="1.0" encoding="UTF-8"?><XMLPayRequest version="2"
Timeout="40"><RequestData><Vendor>6</Vendor><Partner>101</Partner><Transactions><Tr
ansaction><Sale><PayData><Invoice><BillTo><Address><Street> 12115
LACKLAND</Street><Zip>631461234</Zip></Address></BillTo><ShipTo><Address><Zip>6314
61234</Zip></Address></ShipTo><TotalAmt>29.95</TotalAmt></Invoice><Tender><Card><Car
dType>VISA</CardType><CardNum>4012000033330026</CardNum><ExpDate>0408</ExpDat
e><Amount>29.95</Amount><ExtData>CVPresence=Illegible</ExtData></Card></Tender></Pa
yData></Sale></Transaction></Transactions></RequestData><RequestAuth><UserPass><User
>fdcnorth</User><Password>123</Password></UserPass></RequestAuth></XMLPayRequest>
```

## ProcessDebitCard

This Web service operation processes debit card transactions for a merchant. The URL to access this Web service is:

<https://localhost/SmartPayments/transact.asmx?op=ProcessDebitCard>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter | Value |
|-----------|-------|
|-----------|-------|

|                      |  |
|----------------------|--|
| <b>UserName</b>      | Required. User name assigned in the payment server   |
| <b>Password</b>      | Required. Password for the user name assigned in the payment server  |
| <b>TransType</b>     | <p>Required. Type of the debit card transaction. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>Sale</b> to make a purchase on a debit card</li> <li>• <b>Return</b> to credit the cardholder's account</li> <li>• <b>Auth</b> to authorize an amount on a debit card. Pertains only to Concord EFS fuel transactions</li> <li>• <b>Force</b> to place <b>Auth</b> transactions into the current batch (PostAuth). Pertains only to Concord EFS fuel transactions</li> <li>• <b>Capture</b> to settle a single transaction in the current batch; only for terminal-based processors</li> <li>• <b>CaptureAll</b> to settle all transactions in the current batch; only for terminal-based processors or host-based processors that support a batch release feature</li> </ul> |
| <b>CardNum</b>       | Required except for <b>Capture</b> and <b>CaptureAll</b> . Debit card number to process the transaction  |
| <b>ExpDate</b>       | Required except for <b>Capture</b> and <b>CaptureAll</b> . Debit card's expiration date in MYY format  |
| <b>MagData</b>       | <p>Required except for <b>Capture</b> and <b>CaptureAll</b>; required for all swiped card transactions. Data located on the track 2 of the magnetic strip of the card. Once this field is populated, the transaction will be indicated as <i>Card-Present</i> transaction and usually result in a more favorable retail discount rate. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</p> <p>The format of the MagData (or Track 2 data) is CardNum=ExpDate followed by the service code and checksum. For example,<br/>36438999960016=05121015432112345678</p>  |
| <b>NameOnCard</b>    | Optional, depending on different merchant processor setup. The cardholder's name as it appears on the card. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details   |
| <b>Amount</b>        | Required except for <b>CaptureAll</b> . The total transaction amount in DDDD.CC format. This amount includes <b>CashBackAmt</b> and <b>SureChargeAmt</b>   |
| <b>InvNum</b>        | Optional. Invoice tracking number. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details  |
| <b>PNRef</b>         | Optional except for <b>Force</b> and <b>Capture</b> . The reference number assigned by the payment server  |
| <b>Pin</b>           | Required except for <b>Capture</b> and <b>CaptureAll</b> transactions and PIN-less debit transactions. The encrypted pin block returned by the pin-pad. The transaction will fail if an unencrypted pin value is used  |
| <b>RegisterNum</b>   | Optional. A number uniquely identifies the register or computer on which the transaction is performed. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details  |
| <b>SureChargeAmt</b> | Optional. The amount in DDDD.CC format that a merchant charges for processing a debit card transaction   |

|                    |  |
|--------------------|--|
| <b>CashBackAmt</b> | Optional. The amount in DDDD.CC format that a cardholder requests for cash back  |
| <b>ExtData</b>     | <p>Optional, except for <b>&lt;KeySerialNumber&gt;</b>, which is required for all non-PIN-less <b>Sale, Auth, Force, and Return</b> debit transactions, and <b>&lt;Items&gt;</b> and associated nested data elements (required for Concord EFS fuel purchases- see section below). Extended data in XML format. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>&lt;TimeOut&gt;</b><i>TimeOut</i><b>&lt;/TimeOut &gt;</b> for timeout value in seconds (default = 40)</li> <li>• <b>&lt;TrainingMode&gt;</b><i>TrainingMode</i><b>&lt;/TrainingMode &gt;</b> to process transaction in Training Mode; either <b>T</b> or <b>F</b></li> <li>• <b>&lt;KeySerialNumber&gt;</b><i>KeySerialNumber</i><b>&lt;/ KeySerialNumber &gt;</b> for managing DUKPT pin-pads for non-PIN-less debit transactions</li> <li>• <b>&lt;Force&gt;</b><i>Force</i><b>&lt;/Force &gt;</b> for forcing duplicate transactions to be processed; either <b>T</b> or <b>F</b>. Note that some processors, including Concord EFS, will not utilize this tag and may still reject a duplicate transaction</li> </ul> <p>❖ <b>&lt;Items&gt;</b> Required for Concord EFS fuel purchases. Items included in invoice. Contains one or more <b>&lt;Item&gt;</b> elements</p> <ul style="list-style-type: none"> <li>➤ <b>&lt;Item&gt;</b> Required for Concord EFS fuel transactions of type <b>Sale</b> and <b>Force</b>. One item in invoice (item details nested within). There may be multiple <b>&lt;Item&gt;</b> nested within <b>&lt;Items&gt;</b> tag <ul style="list-style-type: none"> <li>▪ <b>&lt;SKU&gt;</b><i>SKU</i><b>&lt;/SKU &gt;</b> SKU number of item</li> <li>▪ <b>&lt;UPC&gt;</b><i>UPC</i><b>&lt;/UPC &gt;</b> Required for Concord EFS fuel purchases. The NACS (National Association of Convenience Stores) product code for fuel purchases. The NACS is an industry standard list that Concord is utilizing. For a list of NACS product codes, please contact EFSnet™ customer support at <a href="mailto:support@concordefsnet.com">support@concordefsnet.com</a> or 1-877-852-2637.</li> <li>▪ <b>&lt;Description&gt;</b><i>Description</i><b>&lt;/Description &gt;</b> Item description</li> <li>▪ <b>&lt;Quantity&gt;</b><i>Quantity</i><b>&lt;/Quantity &gt;</b> Required for Concord EFS fuel purchases. Quantity purchased of item</li> <li>▪ <b>&lt;UnitOfMeasurement&gt;</b><i>UnitOfMeasure</i><b>&lt;/UnitOfMeasurement &gt;</b> Unit of measurement for item</li> <li>▪ <b>&lt;UnitPrice&gt;</b><i>UnitPrice</i><b>&lt;/UnitPrice &gt;</b> Required for Concord EFS fuel purchases. Unit price of item</li> <li>▪ <b>&lt;DiscountAmt&gt;</b><i>DiscountAmount</i><b>&lt;/DiscountAmt &gt;</b> Possible discount amount on item</li> <li>▪ <b>&lt;TaxAmt&gt;</b><i>TaxAmt</i><b>&lt;/TaxAmt &gt;</b> Possible tax amount on item</li> <li>▪ <b>&lt;TotalAmt&gt;</b><i>TotalAmt</i><b>&lt;/TotalAmt &gt;</b> Total amount of item</li> <li>▪ <b>&lt;Category&gt;</b><i>Category</i><b>&lt;/Category &gt;</b> Required for Concord EFS fuel purchases. The specific value <b>Fuel</b> to designate fuel purchases</li> <li>▪ <b>&lt;TaxRate&gt;</b><i>TaxRate</i><b>&lt;/TaxRate &gt;</b> Possible tax rate applied to item</li> </ul> </li> <li>➤ <b>&lt;/Item &gt;</b></li> </ul> <p>❖ <b>&lt;/Items &gt;</b></p> |

## **PIN-less Debit Transactions**

In some cases, debit transactions can actually be processed without the customer's entering a PIN number (a "PIN-less" debit transaction). Essentially, the same information is sent as in a typical PIN-based debit transaction, with the exception of the encrypted PIN-block and key serial number. This transaction type is currently only available with Concord EFS and Global Payments processors.

So, if the processor is not Concord or Global, then **both** the PIN-block and key serial number are required, and without both pieces of data, a transaction will be rejected at the Payment Server. If the designated processor is Concord or Global, then the transaction will be accepted either with **both** pieces of data (interpreted as a PIN-based debit transaction) **or** accepted with **neither** piece of data (interpreted as a PIN-less debit transaction). See Example 3 below.

After the above requirements are met for a transaction, a PIN-less debit transaction will be allowed through the Payment Server. However, it still must have sufficient information to be accepted as a PIN-less transaction only when Concord is the processor. In order for the proper information to be forwarded to Concord for PIN-less debit (and thus for the transaction to be accepted at the processor), the Payment Server must be configured as described below:

### **Application Id Setup**

To process PIN-less debit through Concord, the Application Id sent to Concord must be specified to identify the application in use. Use the following SQL script to change this value in your database:

```
INSERT INTO [dbo].[AppSetting_T] ([Application_Key], [AppSetting_Key],  
[AppSetting_Value], [XmlProfile_TXT]) VALUES (8, 'CustomAppName', 'Your  
Application Name', '')
```

In the above script, you must change '**Your Application Name**' to the Application ID value Concord is expecting, which is typically your company name. Follow these steps in order to execute this script:

- a) Open Query Analyzer
- b) Set the current database to your server database
- c) Paste the above script into the query editor and change '**Your Application Name**' to your company name
- d) Execute the script and verify its success

The CustomAppName is only sent to Concord for PIN-less debit transactions. If CustomAppName is not specified, then the default Application ID will be sent.

## Register Number and Terminal Id Setup

When processing transactions with Concord, the Payment Server will detect that the register number passed from the client-side application matches the Register Number field setup in the merchant account. Once it has made the match, then it will send the corresponding Terminal ID field set up for that Register Number to Concord. When no Terminal ID field is sent to Concord, it defaults to what is set up at the processor (usually Terminal ID “01”). If you are also doing VRU (phone-originated) transactions, a separate Terminal ID field will need to be set up in the Registers of the merchant account and submitted in your request through the Web Service. However, if the merchant will be doing both Internet and VRU transactions at the same time, the Terminal ID value will be required to differentiate between the two. For example, you may set up “01” for Internet and “02” for VRU, and the request sent through the ProcessDebitCard operation, from the merchant’s PIN-less Debit application, must send the appropriate Register Number to reflect what Terminal ID should be sent.

## Fuel Purchases: Debit Card Use

Debit card processing for fuel purchases is now available, currently through Concord EFS only. This functionality allows for fuel purchases with standard debit cards (Visa, Mastercard, etc). Debit fuel purchases (**TransType**’s **Sale** and **Force**) require item-level purchase information. If all the required information for a certain purchase is not provided, the transaction will be rejected and an error message generated. The main implication for the developer is that additional data must be passed to the gateway in order for fuel purchases to process correctly.

Item-level debit fuel purchase information is passed inside the **<Items>** tag in **ExtData**. Fuel purchases are differentiated at the gateway from other purchases by the **Fuel** designation placed within the **<Category>** tag in item descriptions (see Examples 4 through 6 below). In effect, a transaction will only be treated as a fuel transaction if at least one of the items within **<Items>** is designated as category **Fuel**. See the above chart for details on these and other required XML tags for standard debit card processing on fuel transactions.

Note that PIN information and Key Serial Data must be passed on all debit transactions. This data will *not* be retained after a transaction, so the customer must be present to re-enter the PIN. This is important in the case of a **Force** (PostAuth). See examples 5 and 6 below.

## Examples

The following tables contain sample data you can use to test this Web service. The **UserName**, **Password**, **Pin**, and **KeySerialNumber** parameters should be changed when testing the examples yourself.

**Example 1:** The example data below will process a swiped debit card **Sale** transaction through the payment server.

| Parameter          | Value   |
|--------------------|---|
| <b>UserName</b>    | test  |
| <b>Password</b>    | 123   |
| <b>TransType</b>   | Sale  |
| <b>CardNum</b>     | 4055011111111111  |
| <b>ExpDate</b>     | 1205  |
| <b>MagData</b>     | 4055011111111111=05121015432112345678                                     |
| <b>NameOnCard</b>  | John Doe  |
| <b>Amount</b>      | 1.00  |
| <b>InvNum</b>      | 1001  |
| <b>Pin</b>         | 6366C0466A74C3F6  |
| <b>CashBackAmt</b> | 0.5   |
| <b>ExtData</b>     | <Timeout>100</Timeout><KeySerialNumber>4A003102930003BB</KeySerialNumber> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><Message>APPROVAL
  VITAL7</Message><AuthCode>VITAL7</AuthCode><PNRef>2428</PNRef><Ext
  Data>InvNum=1001,CardType=DEBIT</ExtData>
```

```
</Response>
```

**Example 2:** The example data below will process a swiped debit card **Return** transaction through the payment server. The **PNRef** parameter should be changed when testing this example yourself.

| Parameter        | Value   |
|------------------|---|
| <b>UserName</b>  | test  |
| <b>Password</b>  | 123   |
| <b>TransType</b> | Return  |
| <b>CardNum</b>   | 4055011111111111                                    |
| <b>ExpDate</b>   | 1205  |
| <b>MagData</b>   | 4055011111111111=05121015432112345678               |
| <b>Amount</b>    | 1.00  |
| <b>PNRef</b>     | 2428  |
| <b>Pin</b>       | 6366C0466A74C3F6                                    |
| <b>ExtData</b>   | <KeySerialNumber>4A003102930003BB</KeySerialNumber> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
  <Result>0</Result> <RespMSG>Approved</RespMSG> <Message>APPROVAL
  VITAL9</Message> <AuthCode>VITAL9</AuthCode> <PNRef>2430</PNRef> <Ext
  Data>CardType=DEBIT</ExtData>
</Response>
```

**Example 3:** The example data below will process a swiped PIN-less debit card **Sale** transaction through the payment server. Note that this will currently only work with specific payment processors and will be processed as PIN-less debit when both the PIN-block and key serial number information are purposefully omitted.

| Parameter        | Value            |
|------------------|------------------|
| <b>UserName</b>  | test             |
| <b>Password</b>  | 123              |
| <b>TransType</b> | Sale             |
| <b>CardNum</b>   | 4011190070070071 |
| <b>ExpDate</b>   | 0606             |

|                   |                               |
|-------------------|-------------------------------|
| <b>MagData</b>    | 4011190070070071=060600199100 |
| <b>NameOnCard</b> | John Doe                      |
| <b>Amount</b>     | 1.00                          |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><Message>APPROVAL</Message><AuthCode>216880</AuthCode>
<PNRef>38472</PNRef><HostCode>100008691797</HostCode><ExtData>CardType=DEBIT</ExtData>
```

```
</Response>
```

**Example 4:** The example data below will process a swiped debit card **Sale** transaction as a fuel transaction through the payment server.

| Parameter         | Value  |
|-------------------|--|
| <b>UserName</b>   | test   |
| <b>Password</b>   | 123  |
| <b>TransType</b>  | Sale   |
| <b>CardNum</b>    | 4011190070070071   |
| <b>ExpDate</b>    | 0606   |
| <b>MagData</b>    | 4011190070070071=060600199100  |
| <b>NameOnCard</b> | John Doe   |
| <b>Amount</b>     | 1.00   |
| <b>Pin</b>        | A0C98099B1341075   |
| <b>ExtData</b>    | <KeySerialNumber>1234567890000343</KeySerialNumber><Items><Item><UPC>001</UPC><Quantity>1</Quantity><UnitPrice>1</UnitPrice><Category>Fuel</Category><Description>UnLeaded</Description><UnitOfMeasurement>Gallon</UnitOfMeasurement></Item></Items> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><Message>APPROVAL</Message><AuthCode>245028</AuthCode><PNRef>39548</PNRef><HostCode>10008813318</HostCode><ExtData>CardType=DEBIT</ExtData>

</Response>
```

**Example 5:** The example data below will process a swiped debit card **Auth** transaction through the payment server.

| Parameter         | Value   |
|-------------------|---|
| <b>UserName</b>   | test  |
| <b>Password</b>   | 123   |
| <b>TransType</b>  | Auth  |
| <b>CardNum</b>    | 4011190070070071                                    |
| <b>ExpDate</b>    | 0606  |
| <b>MagData</b>    | 4011190070070071=060600199100                       |
| <b>NameOnCard</b> | John Doe  |
| <b>Amount</b>     | 50.00   |
| <b>Pin</b>        | A0C98099B1341075                                    |
| <b>ExtData</b>    | <KeySerialNumber>1234567890000343</KeySerialNumber> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>

<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">

<Result>0</Result><Message>APPROVAL</Message><AuthCode>245267</AuthCode><PNRef>39549</PNRef><HostCode>10008813334</HostCode><ExtData>CardType=DEBIT</ExtData>

</Response>
```

**Example 6:** The example data below will process a swiped debit card **Force** transaction (after Example 5 **Auth**) as a fuel transaction through the payment server. The **PNRef** parameter should be changed when testing this example yourself.

| Parameter        | Value            |
|------------------|------------------|
| <b>UserName</b>  | test             |
| <b>Password</b>  | 123              |
| <b>TransType</b> | Force            |
| <b>CardNum</b>   | 4011190070070071 |

|                   |  |
|-------------------|--|
| <b>ExpDate</b>    | 0606   |
| <b>MagData</b>    | 4011190070070071=060600199100  |
| <b>NameOnCard</b> | John Doe   |
| <b>Amount</b>     | 50.00  |
| <b>PNRef</b>      | 39549  |
| <b>Pin</b>        | A0C98099B1341075   |
| <b>ExtData</b>    | <KeySerialNumber>1234567890000343</KeySerialNumber><Items><Item><UPC>001</UPC><Quantity>1</Quantity><UnitPrice>1</UnitPrice><Category>Fuel</Category><Description>UnLeaded</Description><UnitOfMeasurement>Gallon</UnitOfMeasurement></Item></Items> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><Message>APPROVAL</Message><AuthCode>245267</AuthCode><PNRef>39554</PNRef><HostCode>100008813347</HostCode><ExtData>CardType=DEBIT</ExtData>
```

```
</Response>
```

## ProcessEBTCard

This Web service operation processes EBT card transactions for a merchant. The URL to access this service is:

<https://localhost/SmartPayments/transact.asmx?op=ProcessEBTCard>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter        | Value  |
|------------------|--|
| <b>UserName</b>  | Required. User name assigned in the payment server   |
| <b>Password</b>  | Required. Password for the user name assigned in the payment server  |
| <b>TransType</b> | Required. Type of the EBT card transaction. Valid values are: <ul style="list-style-type: none"> <li><b>FoodStampSale</b> to make a purchase on an EBT cardholder’s food stamp account</li> <li><b>FoodStampReturn</b> to credit to an EBT cardholder’s food stamp account</li> <li><b>CashBenefitSale</b> to make a purchase on an EBT cardholder’s cash benefit account</li> <li><b>Inquire</b> to check the balance on an EBT card</li> </ul> |

|                      |  |
|----------------------|--|
|                      | <ul style="list-style-type: none"> <li>• <b>Capture</b> to settle a single transaction in the current batch; only for terminal-based processors</li> <li>• <b>CaptureAll</b> to settle all transactions in the current batch; only for terminal-based processors or host-based processors that support a batch release feature</li> </ul>  |
| <b>CardNum</b>       | Required except for <b>Capture</b> and <b>CaptureAll</b> . EBT card number to process the transaction  |
| <b>ExpDate</b>       | Required except for <b>Capture</b> and <b>CaptureAll</b> . EBT card's expiration date in MMY format  |
| <b>MagData</b>       | <p>Optional. Data located on the track 2 of the magnetic strip of the card. Once this field is populated, the transaction will be indicated as <i>Card-Present</i> transaction and usually result in a favorable retail discount rate. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</p> <p>The format of the MagData (or Track 2 data) is CardNum=ExpDate followed by the service code and checksum. For example,<br/>36438999960016=05121015432112345678</p>  |
| <b>NameOnCard</b>    | Optional, depending on different merchant processor setup. The cardholder's name as it appears on the card   |
| <b>Amount</b>        | Required except for <b>CaptureAll</b> . The total transaction amount in DDDD.CC format. This amount includes <b>CashBackAmt</b> and <b>SureChargeAmt</b>   |
| <b>InvNum</b>        | Optional. Invoice tracking number. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details  |
| <b>PNRef</b>         | Optional except for <b>FoodStampReturn</b> and <b>Capture</b> . The reference number assigned by the payment server  |
| <b>Pin</b>           | Required except for <b>Capture</b> and <b>CaptureAll</b> . The encrypted pin block returned by the pin-pad. The transaction will fail if an unencrypted pin value is used  |
| <b>RegisterNum</b>   | Optional. A number uniquely identifies a register or computer, on which the transaction is performed. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details   |
| <b>SureChargeAmt</b> | Optional. The amount in DDDD.CC format that a merchant charges for processing an EBT card transaction  |
| <b>CashBackAmt</b>   | Optional. The amount in DDDD.CC format that a cardholder requests for cash back. If used, only good for <b>TransType</b> of <b>CashBenefitSale</b>   |
| <b>ExtData</b>       | <p>Optional except for <b>&lt;KeySerialNumber&gt;</b>, which is required for <b>FoodStampSale</b>, <b>FoodStampReturn</b>, <b>CashBenefitSale</b>, and <b>Inquire</b> with DUKPT pin-pad setup. Extended data in XML format. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>&lt;TimeOut&gt;</b><i>TimeOut</i><b>&lt;/TimeOut&gt;</b> for timeout value in seconds (default = 40)</li> <li>• <b>&lt;TrainingMode&gt;</b><i>TrainingMode</i><b>&lt;/TrainingMode&gt;</b> to process transaction in Training Mode; either <b>T</b> or <b>F</b></li> <li>• <b>&lt;KeySerialNumber&gt;</b><i>KeySerialNumber</i><b>&lt;/KeySerialNumber &gt;</b> for managing DUKPT pin-pads for EBT transactions</li> <li>• <b>&lt;Force&gt;</b><i>Force</i><b>&lt;/Force&gt;</b> for forcing duplicate transactions to be processed; either <b>T</b> or <b>F</b>. Note that some processors, including Concord EFS,</li> </ul> |

|  |  |
|--|--|
|  | will not utilize this tag and may still reject a duplicate transaction |
|--|--|

The following tables contain sample data you can use to test this Web service. The **UserName**, **Password**, **Pin**, and **KeySerialNumber** parameters should be changed when testing the examples yourself.

## Examples

**Example 1:** The example data below will process a swiped EBT card **FoodStampSale** transaction through the payment server.

| Parameter        | Value   |
|------------------|---|
| <b>UserName</b>  | test  |
| <b>Password</b>  | 123   |
| <b>TransType</b> | FoodStampSale                                       |
| <b>CardNum</b>   | 4055011111111111                                    |
| <b>ExpDate</b>   | 1205  |
| <b>MagData</b>   | 4055011111111111=05121015432112345678               |
| <b>Amount</b>    | 10.00   |
| <b>Pin</b>       | 6366C0466A74C3F6                                    |
| <b>ExtData</b>   | <KeySerialNumber>4A003102930003BB</KeySerialNumber> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><Message>FoodStampBa
lanceAmount:
6543.21</Message><AuthCode>VITAL1</AuthCode><PNRef>2431</PNRef><Ex
tData>CardType=EBT</ExtData>
```

```
</Response>
```

**Example 2:** The example data below will process a manually entered EBT card **FoodStampReturn** transaction through the payment server. The **PNRef** parameter should be changed when testing this example yourself.

| Parameter        | Value   |
|------------------|---|
| <b>UserName</b>  | test  |
| <b>Password</b>  | 123   |
| <b>TransType</b> | FoodStampReturn                                     |
| <b>CardNum</b>   | 4055011111111111                                    |
| <b>ExpDate</b>   | 1205  |
| <b>Amount</b>    | 10.00   |
| <b>PNRef</b>     | 2459  |
| <b>Pin</b>       | 6366C0466A74C3F6                                    |
| <b>ExtData</b>   | <KeySerialNumber>4A003102930003BB</KeySerialNumber> |

The following is the result from using the Web service with the above sample data.

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><Message>
  FoodStampBalanceAmount:
  6543.21</Message><AuthCode>VITAL6</AuthCode><PNRef>2460</PNRef><ExtData>CardType=EBT</ExtData>
```

```
</Response>
```

**Example 3:** The example data below will process a manually entered EBT card **CashBenefitSale** transaction through the payment server.

| Parameter          | Value   |
|--------------------|---|
| <b>UserName</b>    | test  |
| <b>Password</b>    | 123   |
| <b>TransType</b>   | CashBenefitSale                                     |
| <b>CardNum</b>     | 4055011111111111                                    |
| <b>ExpDate</b>     | 1205  |
| <b>Amount</b>      | 10.00   |
| <b>InvNum</b>      | 1002  |
| <b>Pin</b>         | 6366C0466A74C3F6                                    |
| <b>CashBackAmt</b> | 5   |
| <b>ExtData</b>     | <KeySerialNumber>4A003102930003BB</KeySerialNumber> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><RespMSG>Approved</RespMSG><Message>CashBenefitB
alanceAmount:
1234.56</Message><AuthCode>VITAL8</AuthCode><PNRef>2461</PNRef><Ex
tData>CardType=EBT</ExtData>
```

```
</Response>
```

## ***ProcessGiftCard***

This Web service operation processes gift card transactions for a merchant. The URL to access this Web service is:

<https://localhost/SmartPayments/transact.asmx?op=ProcessGiftCard>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter        | Value   |
|------------------|---|
| <b>UserName</b>  | Required. User name assigned in the payment server  |
| <b>Password</b>  | Required. Password for the user name assigned in the payment server   |
| <b>TransType</b> | Required. Type of the gift card transaction. Valid values are: <ul style="list-style-type: none"><li>• <b>Redeem</b> to make a purchase on a gift card</li><li>• <b>Reload</b> to increase the balance on a gift card</li><li>• <b>Refund</b> to refund money back to a gift card</li><li>• <b>Activate</b> to activate a gift card</li><li>• <b>Deactivate</b> to deactivate a gift card</li><li>• <b>Inquire</b> to check the balance on a gift card</li><li>• <b>Void</b> to undo an unsettled transaction</li><li>• <b>Force</b> to place a transaction not processed through the payment server into the current batch (ForceAuth). Currently only available through Paymentech</li><li>• <b>Capture</b> to settle a single transaction in the current batch; only for terminal-based processors</li><li>• <b>CaptureAll</b> to settle all transactions in the current batch or host-based processors that support a batch release feature</li></ul> |
| <b>CardNum</b>   | Required. Gift card number used to process the transaction  |
| <b>ExpDate</b>   | Required. Gift card's expiration date in MMY format   |

|                |   |
|----------------|---|
| <b>MagData</b> | <p>Optional except when processing swiped card transactions. Data located on the track 2 of the magnetic strip of the card. Once this field is populated, the transaction will be indicated as <i>Card-Present</i> transaction and usually result in a more favorable retail discount rate. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</p> <p>The format of the MagData (or Track 2 data) is CardNum=ExpDate followed by the service code and checksum. For example,<br/>36438999960016=05121015432112345678</p>  |
| <b>Amount</b>  | Required except for <b>TransType</b> with <b>Inquire</b> . The total transaction amount in DDDD.CC format   |
| <b>InvNum</b>  | Optional. Invoice tracking number. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details   |
| <b>PNRef</b>   | Optional except for TransType with <b>Void</b> . Reference number assigned by the payment server of a previous gift card transaction  |
| <b>ExtData</b> | <p>Optional. Extended data in XML format. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>&lt;TrainingMode&gt;TrainingMode&lt;/TrainingMode&gt;</b> for Training Mode in either <b>T</b> or <b>F</b></li> <li>• <b>&lt;Force&gt;Force&lt;/Force&gt;</b> for forcing duplicate transactions to be processed; either <b>T</b> or <b>F</b>. Note that some processors, including Concord EFS, will not utilize this tag and may still reject a duplicate transaction</li> <li>• <b>&lt;TimeOut&gt;TimeOut&lt;/TimeOut&gt;</b> for timeout value in seconds (default = 40)</li> <li>• <b>&lt;RegisterNum&gt;RegisterNum&lt;/RegisterNum&gt;</b> for register number. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> <li>• <b>&lt;ForceAuth&gt;Force&lt;/ForceAuth&gt;</b> For previously authorized Paymentech transactions of type <b>Redeem</b>, <b>Reload</b>, and <b>Activate</b> to place these transactions into the current batch. (See hierarchy below) Currently only available through Paymentech. See Examples 2 and 3 below</li> <li>• <b>&lt;AuthCode&gt;AuthCode&lt;/AuthCode&gt;</b> Required for Paymentech transactions of type <b>Force (Redeem ForceAuth)</b> See hierarchy below. The authorization code previously obtained for the transaction. Currently only available through Paymentech</li> </ul> <ul style="list-style-type: none"> <li>❖ <b>&lt;ForceAuth&gt;</b> <ul style="list-style-type: none"> <li>➤ <b>&lt;AuthCode&gt;AuthCode&lt;/AuthCode&gt;</b> Required for Paymentech transactions being processed using <b>&lt;ForceAuth&gt;</b> (placed within <b>&lt;ForceAuth&gt;</b> tag). The authorization code previously obtained for the transaction. Currently only available through Paymentech</li> </ul> </li> <li>❖ <b>&lt;/ForceAuth&gt;</b></li> </ul> |

### Gift Card ForceAuth Transactions

When processing gift cards through Paymentech, it is possible to do a transaction of type **Force** (a **Redeem ForceAuth**, for example after a phone authorization is obtained for a purchase) in order to place a gift card **Redeem** transaction in the current batch. Such a

transaction requires **<AuthCode>** to be passed in ExtData (See above table and see Example 2 below).

It is also possible to place previously authorized transactions of type **Redeem, Reload,** and **Activate** in the current batch using the **<ForceAuth>** tag with **<AuthCode>** nested inside (See above table and see Examples 3 and 4 below). Essentially, that means that a **Redeem** can be forced into the batch in two different ways, whereas the **Reload** and **Activate** types must be placed in the batch through the second method. Please note that Gift Card ForceAuth transactions are only required to place a previously authorized **Redeem, Reload,** or **Activate** in the current batch.

## Examples

The following tables contain sample data to process gift card transactions through the payment server. The **UserName** and **Password** parameters should be changed when testing the example yourself.

**Example 1:** The example data below will process a **Redeem** transaction on a gift card through the payment server.

| Parameter        | Value                                    |
|------------------|--|
| <b>UserName</b>  | test                                     |
| <b>Password</b>  | 123                                      |
| <b>TransType</b> | Redeem                                   |
| <b>CardNum</b>   | 6032250001350000156                      |
| <b>ExpDate</b>   | 0509                                     |
| <b>MagData</b>   | 6032250001350000156=09051015432112345678 |
| <b>Amount</b>    | 10.00                                    |
| <b>InvNum</b>    | 1001                                     |
| <b>ExtData</b>   | <Force>T</Force>                         |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><Message>GiftCardBalanceAmount:
  10.00</Message><PNRef>2355</PNRef><HostCode>100004389913</HostCode
  ><ExtData>InvNum=1001,CardType=EGC</ExtData>
```

</Response>

**Example 2:** The example data below will process a swiped gift card **Force** transaction through the payment server. The AuthCode value should be changed when testing this example yourself.

| Parameter         | Value                             |
|-------------------|-----------------------------------|
| <b>UserName</b>   | test                              |
| <b>Password</b>   | 123                               |
| <b>TransType</b>  | Force                             |
| <b>CardNum</b>    | 6035718888880552378               |
| <b>ExpDate</b>    | 1210                              |
| <b>MagData</b>    | 6035718888880552378=1012000876414 |
| <b>NameOnCard</b> | John Doe                          |
| <b>Amount</b>     | 3.00                              |
| <b>ExtData</b>    | <AuthCode>104013</AuthCode>       |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://localhost/SmartPayments/">
```

```
<Result>0</Result><Message>APPROVED</Message><AuthCode>104013</AuthCode><PNRef>39782</PNRef><HostCode>00000014</HostCode><ExtData>CardType=EGC</ExtData>
```

</Response>

**Example 3:** The example data below will place a gift card **Redeem** transaction into the current batch. The AuthCode value should be changed when testing this example yourself.

| Parameter        | Value               |
|------------------|---------------------|
| <b>UserName</b>  | test                |
| <b>Password</b>  | 123                 |
| <b>TransType</b> | Redeem              |
| <b>CardNum</b>   | 6035718888880552378 |

|                   |  |
|-------------------|--|
| <b>ExpDate</b>    | 1210   |
| <b>NameOnCard</b> | John Doe   |
| <b>Amount</b>     | 3.00   |
| <b>ExtData</b>    | <ForceAuth><AuthCode>105258</AuthCode></ForceAuth> |

<?xml version="1.0" encoding="utf-8" ?>

= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://localhost/SmartPayments/">

<Result>0</Result><Message>APPROVED</Message><AuthCode>105259</AuthCode><PNRef>39783</PNRef><HostCode>0000015</HostCode><ExtData>CardType=EGC</ExtData>

</Response>

**Example 4:** The example data below will place a gift card **Reload** transaction into the current batch. The AuthCode value should be changed when testing this example yourself.

| Parameter         | Value  |
|-------------------|--|
| <b>UserName</b>   | test   |
| <b>Password</b>   | 123  |
| <b>TransType</b>  | Reload   |
| <b>CardNum</b>    | 6035718888880552378                                |
| <b>ExpDate</b>    | 1210   |
| <b>NameOnCard</b> | John Doe   |
| <b>Amount</b>     | 20.00  |
| <b>ExtData</b>    | <ForceAuth><AuthCode>105260</AuthCode></ForceAuth> |

<?xml version="1.0" encoding="utf-8" ?>

= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://localhost/SmartPayments/">

<Result>0</Result><Message>APPROVED</Message><AuthCode>105261</AuthCode><PNRef>39784</PNRef><HostCode>0000016</HostCode><ExtData>CardType=EGC</ExtData>

</Response>

## ProcessLoyaltyCard

This Web service operation processes loyalty card transactions for a merchant. The URL to access this Web service is:

<https://localhost/SmartPayments/transact.asmx?op=ProcessLoyaltyCard>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter        | Value   |
|------------------|---|
| <b>UserName</b>  | Required. User name assigned in the payment server  |
| <b>Password</b>  | Required. Password for the user name assigned in the payment server   |
| <b>TransType</b> | <p>Required. Type of the gift card transaction. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>Redeem</b> to make a purchase on a gift card</li> <li>• <b>Reload</b> to increase the balance on a gift card</li> <li>• <b>Refund</b> to refund money back to a gift card</li> <li>• <b>Activate</b> to activate a gift card</li> <li>• <b>Deactivate</b> to deactivate a gift card</li> <li>• <b>Inquire</b> to check the balance on a gift card</li> <li>• <b>Void</b> to undo an unsettled transaction</li> <li>• <b>Force</b> to place a transaction not processed through the payment server into the current batch (ForceAuth). Currently only available through Paymentech</li> <li>• <b>Capture</b> to settle a single transaction in the current batch; only for terminal-based processors</li> <li>• <b>CaptureAll</b> to settle all transactions in the current batch or host-based processors that support a batch release feature</li> </ul> |
| <b>CardNum</b>   | Required. Gift card number used to process the transaction  |
| <b>ExpDate</b>   | Required. Gift card’s expiration date in MMY format   |
| <b>MagData</b>   | <p>Optional except when processing swiped card transactions. Data located on the track 2 of the magnetic strip of the card. Once this field is populated, the transaction will be indicated as <i>Card-Present</i> transaction and usually result in a more favorable retail discount rate. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</p> <p>The format of the MagData (or Track 2 data) is CardNum=ExpDate followed by the service code and checksum. For example,<br/>36438999960016=05121015432112345678</p>  |
| <b>Amount</b>    | Required except for <b>TransType</b> with <b>Inquire</b> . The total transaction amount in DDDD.CC format   |
| <b>InvNum</b>    | Optional. Invoice tracking number. This parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details   |
| <b>PNRef</b>     | Optional except for TransType with <b>Void</b> . Reference number assigned by the payment server of a previous gift card transaction  |

|                |  |
|----------------|--|
| <b>ExtData</b> | <p>Optional. Extended data in XML format. Valid values are:</p> <ul style="list-style-type: none"> <li>• <b>&lt;TrainingMode&gt;</b><i>TrainingMode</i><b>&lt;/TrainingMode&gt;</b> for Training Mode in either <b>T</b> or <b>F</b></li> <li>• <b>&lt;Force&gt;</b><i>Force</i><b>&lt;/Force&gt;</b> for forcing duplicate transactions to be processed; either <b>T</b> or <b>F</b>. Note that some processors, including Concord EFS, will not utilize this tag and may still reject a duplicate transaction</li> <li>• <b>&lt;TimeOut&gt;</b><i>TimeOut</i><b>&lt;/TimeOut&gt;</b> for timeout value in seconds (default = 40)</li> <li>• <b>&lt;RegisterNum&gt;</b><i>RegisterNum</i><b>&lt;/RegisterNum&gt;</b> for register number. The data within this XML tag parameter will remove invalid characters. See list of <a href="#">Removed Characters</a> for more details</li> </ul> |
|----------------|--|

## Examples

This example shows a redeem transaction for Loyalty cards.

| Parameter                             | Value  |
|---------------------------------------|--|
| UserName:                             | <input type="text" value="smart"/>                     |
| Password:                             | <input type="text" value="1234"/>                      |
| TransType:                            | <input type="text" value="Redeem"/>                    |
| CardNum:                              | <input type="text" value="6043990501000200"/>          |
| ExpDate:                              | <input type="text" value="1212"/>                      |
| MagData:                              | <input type="text" value="6043990501000200=12129881"/> |
| Amount:                               | <input type="text" value="1"/>                         |
| InvNum:                               | <input type="text"/>                                   |
| PNRef:                                | <input type="text"/>                                   |
| ExtData:                              | <input type="text"/>                                   |
| <input type="button" value="Invoke"/> |  |

Result:

```

<?xml version="1.0" encoding="utf-8" ?>
- <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="SmartPayments">
<Result>0</Result>

```

<Message>LoyaltyCardBalanceAmount: 953</Message>

<AuthCode>A</AuthCode>

<PNRef>27120</PNRef>

<ExtData>CardType=LOYALTY</ExtData>

</Response>

2. This example shows the Reload function where I am adding 5 points to the loyalty card.

| Parameter  | Value  |
|------------|--|
| UserName:  | <input type="text" value="smart"/>                     |
| Password:  | <input type="text" value="1234"/>                      |
| TransType: | <input type="text" value="Reload"/>                    |
| CardNum:   | <input type="text" value="6043990501000200"/>          |
| ExpDate:   | <input type="text" value="1212"/>                      |
| MagData:   | <input type="text" value="6043990501000200=12129881"/> |
| Amount:    | <input type="text" value="5"/>                         |
| InvNum:    | <input type="text"/>                                   |
| PNRef:     | <input type="text"/>                                   |
| ExtData:   | <input type="text"/>                                   |

Result:

<?xml version="1.0" encoding="utf-8" ?>

= <Response xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="SmartPayments">

<Result>0</Result>

<Message>LoyaltyCardBalanceAmount: 957</Message>

<AuthCode>A</AuthCode>

<PNRef>27122</PNRef>

<ExtData>CardType=LOYALTY</ExtData>

</Response>

## ProcessSignature

Signature capture can be accomplished by using the SmartPayments OCX control or this Web service hosted on the payment server. The URL to access this Web service is: <https://localhost/SmartPayments/transact.asmx?op=ProcessSignature>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter            | Description   |
|----------------------|---|
| <b>UserName</b>      | Required. User name assigned in the payment server  |
| <b>Password</b>      | Required. Password for the user name assigned in the payment server   |
| <b>SignatureType</b> | Required. The type of signature to capture. Valid values are: <ul style="list-style-type: none"><li>• <b>Signature1</b> for Lipman credit</li><li>• <b>Signature2</b> for Lipman check</li><li>• <b>Signature3</b> for handheld application using AppForge</li><li>• <b>Signature4</b> for application not using AppForge</li><li>• <b>Receipt1</b> for TIFF file</li></ul>   |
| <b>SignatureData</b> | Required. If the SignatureType is set to <b>Signature4</b> , a string value of vector coordinates delimited with a ^ character in the following format:<br>$x1,y1^x2,y2^xN,yN^~$<br><br>If there is a pen-up event, then you use the coordinate 0,65535 to signal a break in the line<br><br>^ is the coordinate delimiter, ~ is the ending delimiter, and a comma (,) is the vector delimiter<br><br>If the SignatureType is set to <b>Receipt1</b> , then you must compress and Base64 encode the image data. See the examples section below for more information |
| <b>PNRef</b>         | Required. The unique payment reference number assigned to the transaction   |
| <b>Result</b>        | Optional. An indicator that specifies if the processed transaction was approved   |
| <b>AuthCode</b>      | Optional. The authorization code from the authorization of a transaction  |
| <b>ExtData</b>       | Optional. Extended data in XML format. Valid values are: <ul style="list-style-type: none"><li>• <b>&lt;TrainingMode&gt;T&lt;/TrainingMode&gt;</b> an indicator that specifies transactions will be processed for local loop back testing</li></ul>   |

|  |  |
|--|--|
|  | <ul style="list-style-type: none"> <li>• <b>&lt;TrainingMode&gt;F&lt;/TrainingMode&gt;</b> an indicator that specifies transactions will not be processed for local loop back testing</li> </ul> |
|--|--|

The following tables contain sample data you can use for this Web service. The **UserName**, **Password**, and **PNRef** parameters should be changed when testing this example yourself. If a transaction already has an image, then the Web service will return “Original Transaction Already Has Receipt.”

**Examples**

**Example 1:** The example data below will draw a little square on the receipt and associate the receipt with the transaction with reference number 9568.

| Parameter            | Value                           |
|----------------------|---------------------------------|
| <b>UserName</b>      | test                            |
| <b>Password</b>      | 123                             |
| <b>PNRef</b>         | 9568                            |
| <b>SignatureType</b> | Signature4                      |
| <b>SignatureData</b> | 20,20^20,30^30,30^30,20^20,20^~ |

Result:

```
Testing Merchant
123
New York, NY 12345

Date: 07/26/04 Time: 09:06:48

ForceCapture:

Transaction #:          9568
Card Type:             Diners
AccNum:  XXXXXXXXXXXX0016
Exp Date:              XX/XX
Entry:                 Manual
Amount:                1.00
Response:  001350

I Agree to Pay Above Total
Amount According to Card
Issuer Agreement (Merchant
Agreement if Credit Voucher)



Signature X.....
```

## Creating a Receipt Image Transaction from a File

It is possible to send a receipt image file through the ProcessSignature web service operation from a client-side application in order to associate it with a transaction. Due to the overall complexity of creating a receipt image with ProcessSignature, here is a general list of steps your client-side application would need to perform in order to send images to the Payment Server.

1. Get image file from hardware device, etc
2. Convert image format from hardware device to the TIFF image format, if it isn't already in that format.
3. It is required that at this point you perform an LZW compression on the image data in the TIFF format to reduce the file size because the Payment Server will only accept image data up to 25KB. Here's some general information about LZW compression for TIFF images:

*Compresses and decompresses without information loss, achieving compression ratios up to 5:1. It may be somewhat slower to compress and decompress than the PackBits scheme.*

4. At this point, it is required to compress the file itself with Zip compression to reduce the file size. Any PKZip-compatible Zip compressor and decompressor will work.

IPWorks ([www.ipworks.com](http://www.ipworks.com)) is a third-party provider of software tools and they have a product called “IPWorks! Zip” that can simplify the programmatic compression because it implements a PKZip-compatible Zip compressor and decompressor: the most common compression used on Windows Platforms

5. Base64 encode the image. This ensures that the binary-based information transported can be converted properly into text-based characters to send in the SignatureData parameter of ProcessSignature

6. Input the compressed/base64 encoded image data into the SignatureData parameter of ProcessSignature, and send it to the Payment Server

**Example 2:** The example data below will save a receipt image to the Payment Server and associate the receipt with the transaction with reference number 3.

| Parameter            | Value  |
|----------------------|--|
| <b>UserName</b>      | test   |
| <b>Password</b>      | 123  |
| <b>PNRef</b>         | 3  |
| <b>SignatureType</b> | Receipt1   |
| <b>SignatureData</b> | UEsDBBQAAAAIAI2MPzKHziuFjggAAAO+AAA/AAAAUHJvZ3JhbSBGaWxicy9Db21tb24gRmlsZXMvU21hcnRQYXltZW50cyBtaGFyZWQvVGVtcFJlY2VpcHQyYm1w7dp/TBvXHQBw82MEO nfhx4TIymoIldL9gUSKOkBR4gaiLFK7dNKm/gFS+GGpVPJSkkWCRC65/KAxVRmQ5Y9tKpPJ H1VXJov+UduZLHRNQGbtKDBpdRJCOMAVnjSOW0Oc7dyPt3d3vmLfe4dCFxam3gWf7/mj9 +6979179+4u9T995phJXqSvF+Hnc/h5OdNkyoD/pKW/1oRdAADSb7/Ey8oMM8wwwzbFR OJbfKRuha9lp3TUHL/YCE2xji2Vg8ky2n32VCXoM0s5xNo969D55Cdysb1Z5UUHsAbC7PCP2y ZcUbaTFjwbcAuhhlmmLxQqcZtY0Lar+kmyuuI/Ic3l1p4mgEiueJ1TM5sQY1ULYIapRqHGqMa j1pMWmlVRZiCwk1o4aBSTPx68yoyY2Iq5WJpRup7tCJGty9yQobzhM448Fn8Hvj60ZuGZM8 TjMYiyWPBYkxLnksSLUdKSYkf6a0ZpUbl+8qIh3iFOM/s0gIyIWM7wfDhLhicesiAUtUjvkGcDIT DAb6yj64oyLpJx6QiRSTt0wmq3KSmQjRIG6wEpS6CURLum2AIKU93RkPvptufKqB5037dfvY 0p7p04btsm3gLDIAyI8xVcTk3Kaw/KrptfhNc8twUU2vzdS+MVR2yko5ok5GkOC27/wkZc7u zBPbdjP3KeS9mmNvS3EV96pH/Zy3ffmKdreOKvajdbeiSrHE/NXB8xHLrYXOOnftNWpZfpgj+ bkuZQodACKoqWeqphQLdVfHliAMAD+AP4CT3LxKcYzpfslZ39bJqbILds+WS12XQKVP0vLLx jKz1oKSzv+35me3KuOEb7O4MizTV1iom/NRMen+MfLKKYusetWlHtJbXpIN715f6iOGLsoNp8 ovadai5QIkavvWnxZgxdL3cqJIWRGjXhCFGukTzmJfNFYS27gBRRXg7qwP8iZrqmnJrrWFPOU AFjUXhq2viZye/wx6fP8+2DWRM/W3xNnZfDU5PifMwvxaBnVax1MvTPG0OqWVOzgfOF379c 7Lkthi4ySykGT83I+gxdcOzvf70lrhQwq+bBktWkA80BGoAaAB6RBLMhX4KUuiin5jgQD0nXC jAQ3u2YGbanjEETPIOWI6cCz04TGhO8NCWnONJDIhZVLiKxqYoFOUWhtnk5OZUYR22dUGak YM2K1OWx2vckLb6NqR0dF88rDiU1eRSNJ00qqUcUgJpVvjDwmyxzAbVNdSf3UFvvUOu5B+ L5OIA5wdKChdPr1VJ4eIZSP6Ci2dfssydxIO1/+N4Lv/oiJLC9Hd2JjNc9Pd11TD9nc1Olonp7 wl1noHp71E1tRf6u2GGGfYtssltjNG15RO2m/t+KLaUtFkr2yQr7b8Cgp+OUYhFe2j7aYvgXxt HjZ3OGgwfFfyWN9AyWXIT8DveROuyJG+KNjNrbwQBv+yyIwXjY1hbPnEup2vZ3DoU3o8YjO c8YD0++xkKMRjPeXjT6bPXo8Z05z3oYnNbbx2yoAbAvHJfjOZbkp6IyhclI04bkMUww76Iztn GSF2LBqquVliKnyltIKdsJXcPr7ZezTYU6H0zTG34wz4cMTOAN8YzQYzT70p6BIT5uzvur/baPl w5K0OxdZeftU++8VR2bjpvMGw9WZG6TjptQ2x5t/uvjGj93hIPHzt9IzeN/ovzoXPG57Y8Izq txbyFYhPTGZai3KPPXTej2z4Eb5Hoy1vkX7LpftIwewxs6TTc1bT0zTTKCIH9zXMMW4avDVJzuD L9JD2Jp188ZeO5WcBfF22a8PumPyeQi8fqWelReXke6WF56jezFdMGmO9Mdcc53GDuaDhr 7FNb6xj7jlooWLVeY1FzWWHxqv6z1P3yl9waMuE90chjhRAK00loVIXuPF7YE3AFQjvqO17x+ JXq6l64m+skSntf/955jbFe2t55nwL2RsozqrFYANqsYre11nfYHDvgJFYnrN4L2jKB0CFNvxgK+J C6gPhPJl0AOF1gv9Z2o+27YdG+c71diSstOYanMild0gwVRibYh+FrCrUawF8L7XxfZYXE3aYnX qIm191YG17drwVIZa22I+F81rK8Ia6wFck9eHtc0pgo/8GT9GRj/36xocP7mmIL7MxTLi3wUT |

2sOwR+OpY8snD1O99fk95bn1+SaTnc1YOGrCoRNegYM+x0J6vvUxDkTqpi8MRepmtMaOc  
ZSeJTwjumVGA2VgUK5JT5mtJN12o+27YdHCC9k2Em/wUhSidPJJ1qBnebdK2/XK3K4umpu  
mx23DXrL4tbLT/PCN7IyX0PtNlo4lix/N0/YOErHYao1VHJ53v4+ztdbT/IEbFfnjqLGMPPiWgEE  
RFNC/IZ0BovWJte+/sR5C39Z08sV7svm3a5mwqQI1Yd4trtwH9NsUxhYqxbIZwPWPY6ynQe  
ytY2LPR1DbJMAjEgBaBKitE9JbTRCNY0xnMcwww3Zg8avHKQD45x5EPkHm0AL9urTtAE1CD  
LXj+tZbDcsE3V9Ojs1oLfk+7ixoRvMp7zf51ng4OqC1b9q+3TYqfRIq29L1wxS8V225c/cS0gb  
WKz2HFvxUlxdpOxtY0rc+KZ6C/8uYFy1THjgFP9MVJJG6yJui7Z+TD53fpH2GGWaYU/GyJ3  
ZeyUvnoCj1qyLEuumWtMtyH7shteKL2C2oM+XboFi2f5oBaDY50+3UG1OAxzBPyaB+PDTbo  
2J8mvCEQKIQQ+dbk5e/o+M+QQAdVPVTzmeS9sYp2N8hqX3Vf6S2ebC5Oti5h4KPj9FYMzB  
3AuCrjsNOOt2PTh4ecm8YsHYWSYCBN/WU5rUulywTKbdIe2R83MHFp+4xLeVxm+aczPbtSa  
QNVyzQxj1zwUpxHpi3LhDuO1vKkZtupxbyeerzSfLnVrbJGicwQkqvF1H8sGbJJ4ACdfWO869  
GTPDNPYfUESBAhQLFAAAAAgAjYw/MofOK4WOCAAACj4AAD8AAAAAAAAAAAAAaAlaBAAAAA  
FByb2dyYW0gRmlsZXMvQ29tbW9uIEZpbGVzL1NtYXJ0UGF5bWVudHMgU2hhcmVkl1RlbXB  
SZWNlaXB0LmJtcFBLBQYAAAAAAQABAG0AAADrCAAAAAA=

Result:

Test Cert Merchant  
123 Any Street  
Redmond WA , 98052

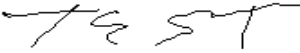
Date: 01/31/05  
Time: 17:36:13

Tran Type: Sale  
Clerk: merchant1  
Invoice:  
ACCT: XXXXXXXXXXXXXXX1111  
EXP: XX/XX  
Issuer: Visa  
CardHolder: John Doe  
Entry Method: Manual  
SYS REF: 3  
Sequence Number: 0353YUJ798GH5E8G5

Result: Approved  
Response: 015245

Am: \$1.00  
Tax: \$0.00  
Tip: \$0.00  
Total: \$1.00

I Agree to Pay Above Total Amount  
According to Card Issuer Agreement  
(Merchant Agreement/Credit Voucher)



Signature X.....

**Example 3:** The example data below will save a check image to the Payment Server and associate it with the transaction with reference number 5454

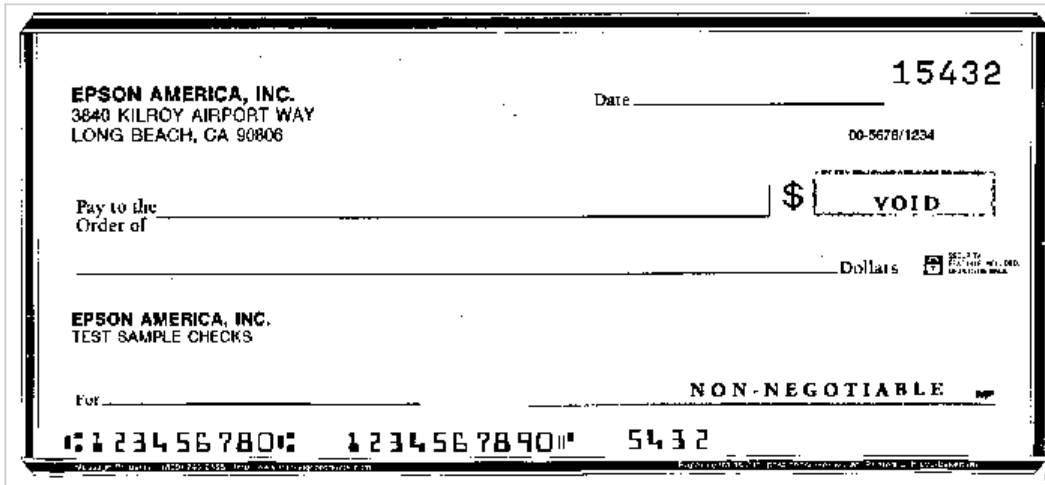
| Parameter       | Value |
|-----------------|-------|
| <b>UserName</b> | test  |
| <b>Password</b> | 123   |
| <b>PNRef</b>    | 5454  |

|                      |   |
|----------------------|---|
| <b>SignatureType</b> | Receipt1  |
| <b>SignatureData</b> | <p>UEsDBBQAAAAIAAJt9zC2UqIIAxYAAEYAAAZAAAAUHJvamVjdHMvU0RLL1RQSSBTb2Z0d2FyZS9FUFNDbGllbnQvVGVTcFJlY2VpcHQuYm1wdVgHXBPZ1gcMEK0jXQKQEKsYKTHFKoksEhRUSy4kbWBjSKIoJBLQIiABBBIXQbIYIsXUCwrCyMgC+uiUj6NAhKpCyIiLDVlZjdB93v7vvd7JzNzz71z59w7c//nf8+Jj4/CYgUFBXX81FFUUCDgpbw4p6j4t65kojhHYc6sjrcoqnzV5RctBSW8MMDPAUXtWZ0qv6U4d1bHbSpSFUI/91Fcqkj+26ZiuKLOVzvymse/xjL8WzdoUpyvoPy1P1vR+P/0dYpWX5/FLSsun7WvimsBirazuh6uBymunNVn5ymX5bYMe4ajA4OxnOHkxJCLvb0j3h6w3NbO3sHRyZkrQKX+rTMZfISqg72drYLCBrYP1T0sMtwG77vWHX+eEfrvtr6Yb/g61mXpa7PchonfKI+ygrEcP6hyRX4oQFb+KiI94ZeErGbKIoqaSzknPWH996tWE/mn5zAOURK6Oj7pMX0jnlHKnlfqz1Iy+4K3bLjgadXhdPkn+0Enz/Kzhc67EyKH+qLohfRCy3TaigVbw5OioB4B88wcTzerpBU6H3QW7CX+etyIHcVhn3yUFO6oGEv+N4ka8osaFEYNJNgbdedubdZmauYwwz0jd2Xz6s1VnfsplzatVfHv0w56xsCnkdB1nqncUda4JVbPcqPLyeE9Amv7gYz0ZIP6PYW7nN0RWD3HQg2+u7UqLeGHvf8juGjrrOvq+nR3MX/giam6FdvT2EyPqRJDp1L98UdW0DFvVXk5+CLA2OyeIk4jsIwBSUwKx03L+BfmABBhOAYpAK2CQvPuG1c68DwewT+ur3R3UlgHRvdatEiqOVtdwOzDM6MSX0pFar3pFF+7Cwu1YfzvrFzoT+qHw+i7aVfQdgcLASEQe0+aEzEQOxCCRIdQAY1c4240rtBT9OQcO1SoYXAgTt3CQUs77VhcBrLR4XotuJy+Dd4QQEKzTO50KQJMAA/asAXsOkMMwbJrNshpfb+Tx/Un1ZA2kh3N7OhfCaNhTbdjZRQ3NxCOajrUQmsG/I+JKQq3NsXS/UXwsKeXkvQZMYv8bOnfr2aOya7eBGawpWIYW2hRaXflw4WaDqhQzbg6TSXtaKQuj1OFmcbCBTjI41Be3p3DGyfes+m87TIOx0eAgePPYDyTggQpisPyyBEj3wRKSnKI2hsF6a9HhiKYLA9BsWgv/0fQCWtd8h2MWZXYIKDzNxxWhpxMeyWhjb5wKPx+q19yqOycRalw3hlgr09MkEfAtk7VA0SO/Itnuf+CCTWIZ9urq3bZ8ZSL4RChFsaDkSavS62DNOFCAzwc0clxw/jrlXTTKPjw7w9rc5oVC3BoJAUvNqTh8MhAm8ybE0AYdDtOn3qPS9HAFvIwMZjIT7PKJ6Xx+Tt8tNZp+Apm800iS6pgp+zsT23HGgr6L+NDWSoJi6kCWdtI6G+4cEMvAvXbLRc1674zYZ5CneZ7le0qlZfMYQvQDZda4IsffQILpB+Lzh7oeAKMSSDL2ZdrYds3qq9F2osOHQyv8kl0ck0IOi0v6xCUhyYj6yu7QxvswWHx5aPjyy1TID5/eYeOo9KVHfPqId/9vZ868nqlwDFa9SG00ngiD8XcrWgfjooZ/hEEndGgQwSq4LRCIfwKoRAY/s4+jkNq2UfDtEniwuEQWI4zPTYUuy0XLk07b0d031Zbp/4VePwDv7iATcWGj3Kmh9nN1LEzrWvvtadoq6WtZ90xZTfzBkxEPFBuS1p0MXZRzD2eIYy7DthagEPI0EUWwMaeEqzqzB8gu/8Bk3gDF4Zvn9h8XIRInEclWiewKyAN2FwvMjf9GfIhKS0j+84C1bteIkToIy4dV8kLvtIWW4ID8h7Dgq4w0/OpOb5Z+fsG7pB67kH5Bmj6v+IBd4+glu3qxIdrhozrA8rFv9bEHLK1y4E1/Y3X5AbHj6vIWL5qvrIp/ZZtBY7s9XSVor2Z/mPNezQkQHx6/GtYsvI7+YXDw9WCrn18A7xYCLUHJSTFqsg05vv6K1curAt2cT7J5HSNcmIoVDJBNfxNeHpGVNLTpIG10/A2NHRTCxIuR7iJfv7oNnVV1Jx0659Pr9H3tzRfg8724pOq1eOQ2msdqDObROx9fdwbARB1v+jn9dREHYzynZCh7nW/OJ6mFCQKisv6j0uN5xX+JKw8gI8IPgyH003wUJkZG2kXcpcf9RCAlm159qcDHOMGfGAIp7Iq62Da1n6fy+/hVCfdMxuLVgt7DzWWIrPzhfWFxu/cfz3+cfHcNtTXUwcdhE1cfZtR2TTrfVGeIDu362fIGTYqpcCcc2nw+S4EOqB96VvNlemqdpMueG89/OYeg0PmEU/oUyGAIEPEXMPsWJ2WZGMpO4rTKC8i6YWRdpCbTcjNZvTCFV8OG/5ABn6ckVdmNfKxYyLvNfbPLjNjCjO7Z42xT8p7CeJETHzfQMxkEDZ7KyWtXrQ4VXQqv4l/1qSuOIk9K64Fetle5mT8sZ0J5mSaNr4drN8Ad1M8Cxpnc1FIlt96WWS5F2Wv1c52q7ZlkmvIzZQNK+jJOI56K9ya3c8zd2W6virmcGUWRLyJ6Z6zwsuCmMHzoqWEmpP1npIs1cyd+7Txe/Y0WnCLVQRlqUGNGQQTWuRmk7K9LfkPZluFu2R7pq5kd3yy1mPWE9Qs8kguSbzP/t3NROf8VJM0rpT9H+KjEwMG6EyougX/Ayr2QkZDMtD4be4hgqJcsF68jrZGODSe7ZmByjX0YUTiZcEHsF6q3k7Ee20t2OJamOITj/qU8gA7XX25B2RL15C4MFSpAVDhEw+jAURBPru5A/BVHsc8ibWk2yCyNIO2a6A6d26271FUqPD2ROgdbOT37SHEMKXEDg7oD+5hn7t3npeiIob28FXACf28xZq3XQOIZtESz9aS46Onz3c9QjivVVmmiHOafg+vMegBVroJdjzuaI7E0hUtuy+V4Mq7KwUYs04tDeuZlVaC7AWb5hJtNnTW6PFNNXdRygzFsrsOVy0hVjbdjTL/40/+7gS/U0wlT8I49n4xGX7ZaudknWihj40m8RXbbMumQraHOAoKSnKhvGSbVD4tkbi2+YdbIm/WpfJKHC/zTLZqJenhw8oixXJsA7GFXaHBYq0oNExnfPLyGQziIh7kOPgk73Pom0twi3xDwK9uNGfjRzw7F52hR2F0K8RnxtzSFyM5oz7xaicq3UokSnOvfVWRzL8+4n+8zavO6Ltpu3ft6nN4lgd2erOo4uwX9RXuhKsCQZWOEovW+CAdc13OUvcwBhWImRSvftkgHHL9IKil4hu7I2xkb0VTG5Or8VBvkRnfm1R1i4eocUlwT7psZffS3pDDdF8Z4Kff18GUZeQ7WwFwd7Qb70AWpYwK03ITrakee7nCoJwUPv3bXXPYW60ikxZkqDkZepJssgszzJR9yKmulnTik+d+12JgIN8YxFRZ602MeNUzqvqJDwFjLjoJpQK5w/+HCEae9uqytVzLIy8o02N6mxPIYwUUD2/5C2Wwj9DrbeVuyUg3h4HPxt1fGJkOLVpCaAmDb6uiQDTJmZdwyZ7jth4kj+3Xdz1e5QSgHNihA0Pi15mnWmMlgIWKRTThNIPAJBSEmevhwCCZrvA3pyXXL4Dv9exTrJJA7BiKRmWkX50iZzaF4KfLtoXi8jmmASNwFqBnAcznVTMAIfh3FtqUtP1a1wzitikGmIdywg+HdReRrEghrCJFEfwbZ973q3vn6qcnw0FW1+MDIdD9hvnNdLiMuSSFWudrOGKKgV4JGch3GpnD0QWegjnMJe3f1UwvrB4/gdE/L4Ds00pzEw1mJPFm0rOoNXcSJKK+IPtNF74/BdH+MRutzo gl1BIhDY1vNOF1YH0P9wtBadbk+Bz+wTCHRv9reqn8WIBokgrhECFvtX86PHjsJpkheIy3K6yE0uV6ZvRkKwVgoRIPFD4BTi56WJ9OXkEI82cvFDcmgBYIPxZ6v9h2bXCWASO3fjBBdkih4XfPf117ykCKOMtc29HA/bAUGW3NG8pXRBO0a0Xp1kqKak7Hf/N3zrFPsiHkBgqV2JefiHbYmVrb59QAxaU0RdFGb23V7IwZf3mmk2Rt22wYLiMI3Uem8i6XMCyyqcHU8j2hB7kjiAbfry25bw2</p> |

YsZp6MiyE8a5/tbmn/73NSc49+xxTX4tx7/oM0sWLDzG62NDU+DGuT0bebIH8C/KudLxGM  
GDcYEyqeQJepORNd6Lsw2gbTBTb6M/21z+brjg/CrJgMxVCAYKi+HcFDgPwDhkmxkquGR+x  
w7O7FvphnpkiWNBIZ5w/vGxvQ9bxuoCAQYueYPb3IYccVuAtz3IWATkIZ4gfnuqQqy3wEzGU  
HHw1tlhgjsCHu1eqjErwji+6KuYqzOOjxOlcaL54PInFaKyMYWpA4arAQzGC3FmFdAdhLBqhIR  
xsR48NigZOYqYDuULFidNkSS+PzIvQ/LpHg6kOu+7w6GImolSWATQqFkuvt3svj23JRMoS  
No8NlzR4SkN+PTCij8Dk1Za4KdktOIrCeI2m9dD++1JGXkzo4C2Y/ttH8fhTgFosDJEI2NZzy0  
AEDaXFBMe9NbpZ0+OI011IkxvP6UqjeviZP+m7WeDn51dR0Vq5scUPeouvJCdrPp3vdMuhK  
vZMWH/BnUE6B8CvtQSVgE/oRoCPJ4xorNjH46wbde5mMKV6d0dlD0+2XVkmB88ULQNpWs  
LHFFFEQR1r4k0YSKt/NbUhm/38y8TzC8X9ploA+Z2etQeGnq/5E/2r8205wF9+2MnRkSIMthp  
AInk9iQ4jpeceIhDPjtiu7RmKTuKjeXD2/wEHF+253r1+OKLxfrVnDhX5meTrMG9PwLAvIln  
5VxmKkHMYqvwldMIF9CE2Q2FpFRH/nZf94VAxT5/PrYKxxC8cjLrv/eB5hhXrUPCplxmS9JpX  
6As+YtiZodumpaYVW5s2tCZ8x8Ffw/umKmHhfc91nNvPtzp74eLEthAXtUbpHX933PvVU7js  
RPQ3Y0GHJaIFXGQ5n6CyJmwxGZCU/wIQo6NNMykku2hFohs1chofd4+iQ5creeqF4GnYVM  
53+zDKFwapLORe+z50QhU3ZL4mNcZ9oFdWjWEWgGFv5QdoMKMSL25M1TjDf7YgNnPMd  
OZCOTJUcetwR86ri0GasJVlfpccs4tdzauVt3ww+Ntba/s2rQ+pumUPwUej33Z2p/LrlanFPPVC  
0mzav/bRRJHVZz0yJTcp56L9zssYpq9d68T7YICnWsjt6H+BDMScbSHwIuyh2e2m6t1kD5Z  
Wvo9Zus1vraTvn8nkPIUC7I3RiLUPQ13M8mfPy2vFjTcZzb00MvelJNg1deZw9zRQGhZzNMx  
GL6ORfXSHoUq2c2hWracgq5uZBVbct0+h0b4x1S6k0Ilca5WhtTt8u+ZNVJWac14j06g3kEw  
2v7Qdnq79IbLe/AZT2fwZpeBZ23p/cpu/Q9aT0VvY3iO5KpQUijRmjV7vrqydP/Tfe7t9kVWw/  
wm7s5GJSw7sibB9Qsbox8ijanXV6vv1kXwXeTA/6MQ6T0m9x+YAA4MJ0Uhd0SOHdyf7vH  
VfRtcTjn7QfHIEhi6DjPKHsVs2zVrPjYkhaTXIORF1EUkzk4e6S1R75L7lsjwPsVEI9773j+vsy0A  
6Tn1PFKrfSNvfwgxcd2FeUZk+1ZfoWtOz/x3cuuDQwgU4+ME3dh7b2t07ry4izcex4lyt0ZcvC  
txsMM5EGK0J9AR37hPXncvT1RNiH8ZqMQnfmYYLZngDo58IUDYBAMnYArPTOUi4akVYAj04  
LAz9zmn0AmaOwA9uL90oxMw17UEb8ffdMAaH/3nq/B7hOd3kZTP37AmtmWOB1v1wIuZX  
2rp6ltLNwGmt7gt6uqMkV6+tUhnFDK8G2oq6o/T42FmRwX6yzIRadFQkhyC9lzsitLS6MXxL  
QPngtX7XbnxzGNouXsKxLB+PuKQFzjesOVZT19AC8MR8f+BHSMAXhohH9nBHxi5N5btQSfz  
xpU7p4zIt7u7721BzdFAOT0wdR9zMMQTWc5NvVfSnjH6xoB9vizMJjHb/VxQEdnSCuz55R9PL  
h5jLTGX83MTI9G/xGr2LFhskqIPaiA+z6bVm7YHFxj8Lj4ZnFqZJm08ZFRh9G2Dg1WLu9gj8  
Yz8k1Ogr5VTFPjngyUYQqy+/VLOW8VNC1537z/rkzGyIO/SffHb9FkVih3Tkr9L9jR3I1jMn  
aNtoZfJc0lofOb4IjFjwUMh1eVP5r06jo4bPVGJZvGu8To7gtj1c0y+KSEpukynKUJL3f+9957oz  
P1LEi5QHDy7PEefBuvQnHeraCp2EbuyjJUjnfmdPfnS1PB9IR5F2W/MDKBYnqQ2ffagJGa4ZxD  
1M2MSaXPvakX4siM7eAmtq3TvPXGij+Wf3htZ/7vr+6Ytvfba7g4vTWNsPn5W93Cmfuvf+44  
Q8WIVLcVYBRuPurtMLkV/5/qoP3L+G5dszX6ejgal6ZuXKonV2ivmLW/imOB9meZ/UHQ9S9Ms  
pZDOChnrIDQty/Xojf/40x4RWb9nHdE/Spfc1ZqQVOi9QVvV3KectajZLMhy35r/ni0kPHtIHc  
ZKi7pOhfZH1tOY5Gry2Grz96l/RhKZy5vWavMIInJpFL5zDtOcSg6kv+pjvT1GuW/Z2CDSOaLs  
fjDR5z+tOgQxdzMj76cvuWuc/KT18WU+giceCXzru5D6ypzprqyxNWq5iwnqRktPxnAHzBnD  
5weVC17XKT35a51pK1yV6nfJxGbjX5JriB5GSyxtTaxoMW8Hs/KYhwuv1QoCnzBpBhFprDKE  
/Skhj085UKW1DH3DE/5Usx6KaPUpYzXRTnJeLSYcswaIWgoH0hMj/rKkjQ647j5+06oEwU9  
dXFiw5UgKap+dtK9Rmta8IC3LIQr622MCZU/oK9OH0txDVSz4VRTIsa1UIIGQcN00UUnuVnv  
BNnr7ffROxOI1WldSjXKm2vJBm04Lc0nkw2Swoce8EpdS96aEdRX6WsTDXmEqs4EJWtQo6  
E/6EqEqEb5c6rpM3wp8ERzTz9pKcmqnrI0J8nwfciULTZC5pHJevLwNPhgXvqSJKGptWWUcy  
5Pn662nydvdoWl1sMlxw2ycaIn06uJxtSjnJc9Vb/WOrdc3xXSrgLStaZp9Bm7z94EM+cI8jBb  
qTMB9a4vTSiB4nPL2/4y6jAkxpQrb7u1yQ2783OI65B7p541nNT5XKCXi+RUvaC6evfH7n8W  
TZPveeK0kc3Nxy5PUS5dyV7Gt9dIOW18FSTd9akUHc2ftbRUU30L0tZdJl7byG11icg8fCT7A5  
vl+JDZPYhw93hbfS7eC4x30Y1cWwv6G1NV5S2HkirOJ9hZ5IQX/WLmsbPNO/Yh+UmHwPj5Nfr  
moLlaEqa5ZgEO1xxXz0XryhoOZGzpY86p7ZrcsDFzP2V8aYRz/277t1o5jF69/QH4C50mRaZQ  
E/mnD3tGppQZ309iUm/zThH1+maVPvqQ5+lgZKJBm/1v/X1BLAQIUCxQAAAAIAAJt9zC2Uqll  
AxYAAEYAAAAzAAAAAAAAAAAAIAAC2gQAAAAABQcm9qZWN0cy9TREsvVFBjJIFNvZnR3YXJlO  
VQU0NsaWVudC9UZw1wUmVjZWlwdC5ibXBQSwUGAAAAAAEAAQbHAAAAVBYAAAAA

Result:

Receipt for this transaction:



## ValidCard

This Web service operation does the validation check on a credit card. It checks the card length based on the card type, performs a mod 10 checksum, and checks the expiration date. The return value could be: 0 – good, 1001 – no card number, 1002 – no expiration date, 1003 – invalid card type, 1004 – invalid card length, 1005 – invalid mod 10 check, 1006 – invalid expiration date. The URL to access this operation is: <https://localhost/SmartPayments/validate.aspx?op=ValidCard>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter         | Description                                    |
|-------------------|--|
| <b>CardNumber</b> | Required. The number of a credit card          |
| <b>ExpDate</b>    | Required. The expiration date of a credit card |

## Examples

The following tables contain sample data you can use to test this Web service.

**Example 1:** The example data below will result in return 0.

| Parameter         | Value            |
|-------------------|------------------|
| <b>CardNumber</b> | 5454545454545454 |
| <b>ExpDate</b>    | 0509             |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<int xmlns="http://localhost/SmartPayments/">0</int>
```

**Example 2:** The example data below will result in return 1005.

| Parameter         | Value            |
|-------------------|------------------|
| <b>CardNumber</b> | 5454545454545453 |
| <b>ExpDate</b>    | 0509             |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<int xmlns="http://localhost/SmartPayments/">1005</int>
```

## ***ValidCardLength***

This Web service operation checks for the card length based on the card type. The URL to access this operation is:

<https://localhost/SmartPayments/validate.aspx?op=ValidCardLength>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter         | Description                           |
|-------------------|---------------------------------------|
| <b>CardNumber</b> | Required. The number of a credit card |

## **Examples**

The following tables contain sample data you can use to test this Web service. The return result will be either true or false.

**Example 1:** The example data below will result in return false.

| Parameter         | Value          |
|-------------------|----------------|
| <b>CardNumber</b> | 54545454545454 |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<boolean xmlns="http://localhost/SmartPayments/">>false</boolean>
```

**Example 2:** The example data below will result in return true.

| Parameter         | Value         |
|-------------------|---------------|
| <b>CardNumber</b> | 4126196901499 |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<boolean xmlns="https://localhost/SmartPayments/">true</boolean>
```

### ***ValidExpDate***

This Web service operation validates the expiration date of a credit card. The URL to access this operation is:

<https://localhost/SmartPayments/validate.aspx?op=ValidExpDate>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter      | Description                                    |
|----------------|--|
| <b>ExpDate</b> | Required. The expiration date of a credit card |

### **Examples**

The following tables contain sample data you can use to test this Web service. The return value will either be true or false.

**Example 1:** The example data below will result in return true.

| Parameter      | Value |
|----------------|-------|
| <b>ExpDate</b> | 0509  |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<boolean xmlns="http://localhost/SmartPayments/">true</boolean>
```

**Example 2:** The example data below will result in return false.

| Parameter      | Value |
|----------------|-------|
| <b>ExpDate</b> | 0304  |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<boolean xmlns="https://localhost/SmartPayments/">false</boolean>
```

## ValidMod10

This Web service operation validates the credit card by performing a mod 10 checksum on the card number. The URL to access this operation is:  
<https://localhost/SmartPayments/validate.aspx?op=ValidMod10>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter         | Description                           |
|-------------------|---------------------------------------|
| <b>CardNumber</b> | Required. The number of a credit card |

## Examples

The following table contains sample data you can use to test this Web service. The return result will be either true or false.

**Example 1:** The example data below will result in return true.

| Parameter         | Value          |
|-------------------|----------------|
| <b>CardNumber</b> | 54545454545454 |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<boolean xmlns="http://localhostSmartPayments/">true</boolean>
```

**Example 2:** The example data below will result in return false.

| Parameter         | Value        |
|-------------------|--------------|
| <b>CardNumber</b> | 545454545454 |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

<boolean xmlns="http://localhost/SmartPayments/">false</boolean>

## AddMerchant

This Web service operation allows you to add a merchant. The URL to access this operation is: <https://localhost/admin/ws/admin.asmx?op=AddMerchant>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter                 | Value  |
|---------------------------|--|
| <b>UserName</b>           | Required. User name assigned in the payment server. This user should have administrative rights to perform this web service. |
| <b>SecureToken</b>        | Required. The secure code given by the payment server located at the Preferences menu, under Password                        |
| <b>ResellerKey</b>        | Optional. The key of the reseller under which the merchant is added  |
| <b>MerchantUsername</b>   | Required. User name for the merchant assigned in the payment server  |
| <b>MerchantPassword</b>   | Required. Password for the merchant name assigned in the payment server  |
| <b>MerchantID</b>         | Optional. ID for the merchant  |
| <b>MerchantID2</b>        | Optional. ID for a second merchant   |
| <b>AnnualSales</b>        | Optional. Annual sales of the company  |
| <b>BusinessStartDate</b>  | Optional. Date the business started  |
| <b>CompanyName</b>        | Optional. Name of the company  |
| <b>DoingBusinessAs</b>    | Optional. Type of company (ex: retail)   |
| <b>Url</b>                | Optional. URL of the company website   |
| <b>FederalTaxID</b>       | Optional. Company's federal tax ID   |
| <b>StateTaxID</b>         | Optional. Company's state tax ID   |
| <b>SalesTaxID</b>         | Optional. Company's sales tax ID   |
| <b>OwnershipType</b>      | Optional. The type of company the merchant belongs to (Ex: corporation)  |
| <b>AutoCloseBatch</b>     | Required. (True or False) Set the close batch automation   |
| <b>AutoCloseBatchHour</b> | Required with True, Optional with False. Set the close batch time  |
| <b>ForceDuplicate</b>     | Required. (True or False) Allows duplicate transaction   |
| <b>RequirePNRef</b>       | Required. (True or False)  |
| <b>ContactFirstName</b>   | Required. First name of the contact person   |
| <b>ContactLastName</b>    | Required. Last name of the contact person  |
| <b>ContactEmail</b>       | Required. Email address of the contact person  |
| <b>ContactDayPhone</b>    | Required. Day time phone number of the contact person  |
| <b>ContactFax</b>         | Optional. Fax number of the contact person   |

|                           |  |
|---------------------------|--|
| <b>ContactStreet1</b>     | Required. Line one of the of the contact person's street address   |
| <b>ContactStreet2</b>     | Optional. Line two of the contact person's street address  |
| <b>ContactCity</b>        | Required. Contact person's city  |
| <b>ContactState</b>       | Required. Contact person's state   |
| <b>ContactPostalCode</b>  | Required. Contact person's zip code  |
| <b>ContactCountryCode</b> | Optional. Contact person's country   |
| <b>TimeZoneOffset</b>     | Optional. Hours off time zone  |
| <b>PaymentMethodXml</b>   | Required. Extended data in XML format  |
| <b>RegistersXml</b>       | Optional. Extended data in XML format  |
| <b>ExtData</b>            | <p>Optional. Extended data in XML format</p> <p>Note: By default, sending the email to the new user/merchant is set to false but the system will send out an email to the merchant by adding this tag:&lt;SendEmail&gt;T&lt;/SendEmail&gt;</p> <p>&lt;VirtualTerminalTipAmount&gt;T or F&lt;/VirtualTerminalTipAmount&gt;<br/>For enabling tip amount to be processed using the virtual terminal.</p> <p>&lt;VirtualTerminalTaxAmount&gt;T or F&lt;/VirtualTerminalTaxAmount&gt; for enabling the virtual terminal tax amount to be processed using the virtual terminal.</p> <p>&lt;VirtualTerminalConvenienceAmount&gt;T or F<br/>    &lt;/VirtualTerminalConvenienceAmount&gt; for enabling the convenience amount to be processed using the virtual terminal.</p> <p>&lt;AutoSettleMerchantEmail&gt;T or F&lt;/AutoSettleMerchantEmail&gt; for enabling the auto settlement merchant email function for the account if auto settle function is enabled on terminal based accounts.</p> |

## Examples

The following table contains sample data you can use to test this Web service. The parameters should be changed when testing this example yourself.

### Example 1: EFSNet

This example table shows the information needed to add a merchant using the EFSNet processor. The parameters should be changed when testing this example yourself.

| Parameter                | Value  |
|--------------------------|--|
| <b>UserName</b>          | admin  |
| <b>SecureToken</b>       | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>=   |
| <b>MerchantUserName</b>  | ex1  |
| <b>MerchantPassword</b>  | 123  |
| <b>AutoCloseBatch</b>    | FALSE  |
| <b>FourceDuplicate</b>   | FALSE  |
| <b>RequirePNRef</b>      | FALSE  |
| <b>ContactFirstName</b>  | John   |
| <b>ContactLastName</b>   | Doe  |
| <b>ContactEmail</b>      | JDoe@test.com  |
| <b>ContactDayPhone</b>   | 555-555-5555   |
| <b>ContactStreet1</b>    | 123 Any St   |
| <b>ContactCity</b>       | Redmond  |
| <b>ContactSate</b>       | WA   |
| <b>ContactPostalCode</b> | 98034  |
| <b>PaymentMethodXml</b>  | <PaymentMethod><ProcessorID>EFSNet</ProcessorID><PaymentTypeID>EGC</PaymentTypeID><HostBased>True</HostBased><XmlProfile><StoreID>richso01</StoreID><StoreKey>A011CD09BB77BD85530F6B2FD3BD8550F62FD377F68798DD1EE0653E6DE5AED2</StoreKey><CurrencyID>USD</CurrencyID><IndustryID>R</IndustryID><DebitSurcharge>0</DebitSurcharge></XmlProfile></PaymentMethod> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
<Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
  <Code>OK</Code>
  <Error>A new Merchant was successfully created!</Error>
  <Partner />
  <Vendor>307</Vendor>
  <Username>ex1</Username>
```

</Result>

## Example 2: FDCN

This example table shows the information needed to add a merchant using the FDCN processor. The parameters should be changed when testing this example yourself.

| Parameter                | Value   |
|--------------------------|---|
| <b>UserName</b>          | admin   |
| <b>SecureToken</b>       | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>=  |
| <b>MerchantUserName</b>  | ex2   |
| <b>MerchantPassword</b>  | 123   |
| <b>AutoCloseBatch</b>    | FALSE   |
| <b>FourceDuplicate</b>   | FALSE   |
| <b>RequirePNRef</b>      | FALSE   |
| <b>ContactFirstName</b>  | John  |
| <b>ContactLastName</b>   | Doe   |
| <b>ContactEmail</b>      | JDoe@test.com   |
| <b>ContactDayPhone</b>   | 555-555-5555  |
| <b>ContactStreet1</b>    | 123 Any St  |
| <b>ContactCity</b>       | Redmond   |
| <b>ContactSate</b>       | WA  |
| <b>ContactPostalCode</b> | 98034   |
| <b>PaymentMethodXml</b>  | <PaymentMethod><ProcessorID>FDCN</ProcessorID><Payment<br>TypeID>MASTERCARD</PaymentTypeID><HostBased>FALSE</H<br>ostBased><XmlProfile><DatawireID>00003F1FF8B71A94485E</<br>DatawireID><MerchantID>00001210566</MerchantID><Termina<br>lID>00001367275</TerminalID><IndustryID>R</IndustryID></<br>XmlProfile></PaymentMethod> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xmlns="http://secure.payment-engine.com/Admin/ws">
```

<Code>OK</Code>

<Error>A new Merchant was successfully created!</Error>

<Partner />

<Vendor>308</Vendor>

<Username>ex2</Username>

</Result>

### Example 3: FDCNorth

This example table shows the information needed to add a merchant using the FDCNorth processor. The parameters should be changed when testing this example yourself.

| Parameter                | Value  |
|--------------------------|--|
| <b>UserName</b>          | Test   |
| <b>SecureToken</b>       | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>=   |
| <b>MerchantUserName</b>  | ex3  |
| <b>MerchantPassword</b>  | 123  |
| <b>AutoCloseBatch</b>    | FALSE  |
| <b>FourceDuplicate</b>   | FALSE  |
| <b>RequirePNRef</b>      | FALSE  |
| <b>ContactFirstName</b>  | John   |
| <b>ContactLastName</b>   | Doe  |
| <b>ContactEmail</b>      | JDoe@test.com  |
| <b>ContactDayPhone</b>   | 555-555-5555   |
| <b>ContactStreet1</b>    | 123 Any St   |
| <b>ContactCity</b>       | Redmond  |
| <b>ContactSate</b>       | WA   |
| <b>ContactPostalCode</b> | 98034  |
| <b>PaymentMethodXml</b>  | <PaymentMethod><ProcessorID>FDCNorth</ProcessorID><PaymentTypeID>DISCOVER</PaymentTypeID><HostBased>FALSE</HostBased><XmlProfile><DatawireID>00000B81C6C0BA03F945</DatawireID><MerchantID>000000925990</MerchantID><Termin |

|  |   |
|--|---|
|  | <pre> alID&gt;462862&lt;/TerminalID&gt; &lt;IndustryID&gt;R&lt;/IndustryID&gt;&lt;CSPhone Number&gt;123-123- 1234&lt;/CSPhoneNumber&gt; &lt;URLEmail&gt;billp@RFNsoft.com&lt;/URLEmail &gt;&lt;DebitSurcharge&gt;0&lt;/DebitSurcharge&gt;&lt;/XmlProfile&gt;&lt;/PaymentMethod&gt; </pre> |
|--|---|

Result:

```

<?xml version="1.0" encoding="utf-8" ?>

- <Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">

<Code>OK</Code>

<Error>A new Merchant was successfully created!</Error>

<Partner />

<Vendor>309</Vendor>

<Username>ex3</Username>

</Result>

```

#### Example 4: FDCSouth

This example table shows the information needed to add a merchant using the FDCSouth processor. The parameters should be changed when testing this example yourself.

| Parameter               | Value   |
|-------------------------|---|
| <b>UserName</b>         | Test  |
| <b>SecureToken</b>      | jYizQ51MNs1wf1fcP25eB3iYRxcg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>= |
| <b>MerchantUserName</b> | ex4   |
| <b>MerchantPassword</b> | 123   |
| <b>AutoCloseBatch</b>   | FALSE   |
| <b>FourceDuplicate</b>  | FALSE   |
| <b>RequirePNRef</b>     | FALSE   |
| <b>ContactFirstName</b> | John  |

|                          |  |
|--------------------------|--|
| <b>ContactLastName</b>   | Doe  |
| <b>ContactEmail</b>      | JDoe@test.com  |
| <b>ContactDayPhone</b>   | 555-555-5555   |
| <b>ContactStreet1</b>    | 123 Any St   |
| <b>ContactCity</b>       | Redmond  |
| <b>ContactState</b>      | WA   |
| <b>ContactPostalCode</b> | 98034  |
| <b>PaymentMethodXml</b>  | <PaymentMethod><ProcessorID>FDCSouth</ProcessorID><PaymentTypeID>AMEX</PaymentTypeID><HostBased>FALSE</HostBased><XmlProfile><DatawireID>0000D900A4BF93DA74CA</DatawireID><MerchantID>67888886496</MerchantID><TerminalID>99</TerminalID><MerchantNumber>67888886496</MerchantNumber><TerminalNumber>99</TerminalNumber><QualCode>25800000</QualCode><TransNumber/><AMEXSENumber>5041079856</AMEXSENumber><JALSENumber>0000000000</JALSENumber><JCBSSENumber>2222222222</JCBSSENumber><DinersSENumber>4444444444</DinersSENumber><CarteBlancheSENumber>5555555555</CarteBlancheSENumber><DiscoverSENumber>1111111111</DiscoverSENumber><IndustryID>E</IndustryID><InputDeviceKey>1</InputDeviceKey><DebitSurcharge>0</DebitSurcharge></XmlProfile></PaymentMethod> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
= <Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
<Code>OK</Code>
<Error>A new Merchant was successfully created!</Error>
<Partner />
<Vendor>310</Vendor>
<Username>ex4</Username>
</Result>
```

**Example 5: InterceptD**

This example table shows the information needed to add a merchant using the InterceptD processor. The parameters should be changed when testing this example yourself.

| Parameter                | Value  |
|--------------------------|--|
| <b>UserName</b>          | Test   |
| <b>SecureToken</b>       | jYizQ51MNs1wf1fcP25eB3iYRxcg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>=  |
| <b>MerchantUserName</b>  | ex5  |
| <b>MerchantPassword</b>  | 123  |
| <b>AutoCloseBatch</b>    | FALSE  |
| <b>FourceDuplicate</b>   | FALSE  |
| <b>RequirePNRef</b>      | FALSE  |
| <b>ContactFirstName</b>  | John   |
| <b>ContactLastName</b>   | Doe  |
| <b>ContactEmail</b>      | JDoe@test.com  |
| <b>ContactDayPhone</b>   | 555-555-5555   |
| <b>ContactStreet1</b>    | 123 Any St   |
| <b>ContactCity</b>       | Redmond  |
| <b>ContactSate</b>       | WA   |
| <b>ContactPostalCode</b> | 98034  |
| <b>PaymentMethodXml</b>  | <PaymentMethod><ProcessorID>InterceptD</ProcessorID><PaymentTypeID>DINERS</PaymentTypeID><HostBased>True</HostBased><XmlProfile><CompanyKey>8990</CompanyKey><PIN>1234</PIN><TerminalID>6177</TerminalID><IndustryID>E</IndustryID><CurrencyID>USD</CurrencyID><DebitSurcharge>0</DebitSurcharge></XmlProfile></PaymentMethod> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
```

```
<Code>OK</Code>
```

```
<Error>A new Merchant was successfully created!</Error>
```

```
<Partner />
```

<Vendor>311</Vendor>

<Username>ex5</Username>

</Result>

### Example 6: PayTampa

This example table shows the information needed to add a merchant using the PayTampa processor. The parameters should be changed when testing this example yourself.

| Parameter                | Value   |
|--------------------------|---|
| <b>UserName</b>          | Test  |
| <b>SecureToken</b>       | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>=  |
| <b>MerchantUserName</b>  | ex6   |
| <b>MerchantPassword</b>  | 123   |
| <b>AutoCloseBatch</b>    | FALSE   |
| <b>ForceDuplicate</b>    | FALSE   |
| <b>RequirePNRef</b>      | FALSE   |
| <b>ContactFirstName</b>  | John  |
| <b>ContactLastName</b>   | Doe   |
| <b>ContactEmail</b>      | JDoe@test.com   |
| <b>ContactDayPhone</b>   | 555-555-5555  |
| <b>ContactStreet1</b>    | 123 Any St  |
| <b>ContactCity</b>       | Redmond   |
| <b>ContactState</b>      | WA  |
| <b>ContactPostalCode</b> | 98034   |
| <b>PaymentMethodXml</b>  | <PaymentMethod><ProcessorID>PayTampa</ProcessorID><PaymentTypeID>CARTBLANCH</PaymentTypeID><HostBased>True</HostBased><XmlProfile><MerchantNumber>70000000544</MerchantNumber><TerminalNumber>002</TerminalNumber><ClientNumber>0002</ClientNumber><UserID>RFNtest1</UserID><Password>bpittman1</Password><HostBased_Manual_Settle>FALSE</HostBased_Manual_Settle><Industry>E</Industry><DebitSurcharge>0</DebitSurcharge></XmlProfile></PaymentMethod> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
  xmlns="http://secure.payment-engine.com/Admin/ws">
```

```
<Code>OK</Code>
```

```
<Error>A new Merchant was successfully created!</Error>
```

```
<Partner />
```

```
<Vendor>312</Vendor>
```

```
<Username>ex6</Username>
```

```
</Result>
```

### Example 7: RMRS

This example table shows the information needed to add a merchant using the RMRS processor. The parameters should be changed when testing this example yourself.

| Parameter               | Value  |
|-------------------------|--|
| <b>UserName</b>         | Test   |
| <b>SecureToken</b>      | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>= |
| <b>MerchantUserName</b> | ex7  |
| <b>MerchantPassword</b> | 123  |
| <b>AutoCloseBatch</b>   | FALSE  |
| <b>ForceDuplicate</b>   | FALSE  |
| <b>RequirePNRef</b>     | FALSE  |
| <b>ContactFirstName</b> | John   |
| <b>ContactLastName</b>  | Doe  |
| <b>ContactEmail</b>     | JDoe@test.com  |
| <b>ContactDayPhone</b>  | 555-555-5555   |
| <b>ContactStreet1</b>   | 123 Any St   |
| <b>ContactCity</b>      | Redmond  |

|                          |  |
|--------------------------|--|
| <b>ContactSate</b>       | WA   |
| <b>ContactPostalCode</b> | 98034  |
| <b>PaymentMethodXml</b>  | <PaymentMethod><ProcessorID>RMRS</ProcessorID><PaymentTypeID>ECHECK</PaymentTypeID><HostBased>True</HostBased><XmlProfile><VerificationPortNumber>54081</VerificationPortNumber><VerificationSiteNumber>641</VerificationSiteNumber><VerificationTerminalNumber>232993</VerificationTerminalNumber><VerificationRuleSet>833</VerificationRuleSet><TruncationPortNumber>54080</TruncationPortNumber><TruncationSiteNumber>65301</TruncationSiteNumber><TruncationTerminalNumber>66</TruncationTerminalNumber><TruncationRuleSet>66</TruncationRuleSet></XmlProfile></PaymentMethod> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
<Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
  <Code>OK</Code>
  <Error>A new Merchant was successfully created!</Error>
  <Partner />
  <Vendor>313</Vendor>
  <Username>ex7</Username>
</Result>
```

### Example 8: VITAL

This example table shows the information needed to add a merchant using the VITAL processor. The parameters should be changed when testing this example yourself.

| Parameter               | Value  |
|-------------------------|--|
| <b>UserName</b>         | Test   |
| <b>SecureToken</b>      | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>= |
| <b>MerchantUserName</b> | ex8  |

|                          |   |
|--------------------------|---|
| <b>MerchantPassword</b>  | 123   |
| <b>AutoCloseBatch</b>    | FALSE   |
| <b>ForceDuplicate</b>    | FALSE   |
| <b>RequirePNRef</b>      | FALSE   |
| <b>ContactFirstName</b>  | John  |
| <b>ContactLastName</b>   | Doe   |
| <b>ContactEmail</b>      | JDoe@test.com   |
| <b>ContactDayPhone</b>   | 555-555-5555  |
| <b>ContactStreet1</b>    | 123 Any St  |
| <b>ContactCity</b>       | Redmond   |
| <b>ContactState</b>      | WA  |
| <b>ContactPostalCode</b> | 98034   |
| <b>PaymentMethodXml</b>  | <pre>&lt;PaymentMethod&gt;&lt;ProcessorID&gt;VITAL&lt;/ProcessorID&gt; &lt;Payment TypeID&gt;DEBIT&lt;/PaymentTypeID&gt; &lt;HostBased&gt;FALSE&lt;/HostBase d&gt; &lt;XmlProfile&gt; &lt;SSL2&gt;True&lt;/SSL2&gt; &lt;ConnectionMethod&gt;SSL2&lt; /ConnectionMethod&gt; &lt;Processor_ID&gt;VITAL&lt;/Processor_ID&gt; &lt;Merc hantNumber&gt;888000000394&lt;/MerchantNumber&gt; &lt;TerminalNumb er&gt;1515&lt;/TerminalNumber&gt; &lt;AgentBankNumber&gt;000000&lt;/Agen tBankNumber&gt; &lt;ChainNumber&gt;111111&lt;/ChainNumber&gt; &lt;StoreNu mber&gt;5999&lt;/StoreNumber&gt; &lt;BinNumber&gt;999995&lt;/BinNumber&gt; &lt;CustomerServicePhone&gt;123- 1231234&lt;/CustomerServicePhone&gt; &lt;LocationNumber&gt;00001&lt;/Lo cationNumber&gt; &lt;PostalCode&gt;98052&lt;/PostalCode&gt; &lt;CategoryCode &gt;5999&lt;/CategoryCode&gt; &lt;VNumber&gt;71004021&lt;/VNumber&gt; &lt;Indu stryID&gt;R&lt;/IndustryID&gt; &lt;TimezoneID&gt;PST&lt;/TimezoneID&gt; &lt;Curre ncyID&gt;840&lt;/CurrencyID&gt; &lt;CountryID&gt;USA&lt;/CountryID&gt; &lt;Langu ageID&gt;00&lt;/LanguageID&gt; &lt;InputDeviceID&gt;2&lt;/InputDeviceID&gt; &lt;A BNumber&gt;11111111&lt;/ABANumber&gt; &lt;SettlementAgentNumber &gt;V001&lt;/SettlementAgentNumber&gt; &lt;SharingGroup&gt;ABC&lt;/Sharing Group&gt; &lt;ReimbursementAttribute&gt;Z&lt;/ReimbursementAttribute&gt; &lt; FCSID /&gt; &lt;DebitSurcharge&gt;0&lt;/DebitSurcharge&gt; &lt;/XmlProfile&gt; &lt;/Paymen tMethod&gt;</pre> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://secure.payment-engine.com/Admin/ws">
```

```
<Code>OK</Code>
```

```
<Error>A new Merchant was successfully created!</Error>
```

```
<Partner />
```

<Vendor>314</Vendor>

<Username>ex8</Username>

</Result>

### Example 9: GlobalEast

This example table shows the information needed to add a merchant using the GlobalEast processor. The parameters should be changed when testing this example yourself.

| Parameter                | Value   |
|--------------------------|---|
| <b>UserName</b>          | Test  |
| <b>SecureToken</b>       | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>=  |
| <b>MerchantUserName</b>  | ex9   |
| <b>MerchantPassword</b>  | 123   |
| <b>AutoCloseBatch</b>    | FALSE   |
| <b>FourceDuplicate</b>   | FALSE   |
| <b>RequirePNRef</b>      | FALSE   |
| <b>ContactFirstName</b>  | John  |
| <b>ContactLastName</b>   | Doe   |
| <b>ContactEmail</b>      | JDoe@test.com   |
| <b>ContactDayPhone</b>   | 555-555-5555  |
| <b>ContactStreet1</b>    | 123 Any St  |
| <b>ContactCity</b>       | Redmond   |
| <b>ContactSate</b>       | WA  |
| <b>ContactPostalCode</b> | 98034   |
| <b>PaymentMethodXml</b>  | <PaymentMethod><ProcessorID>GlobalEast</ProcessorID><PaymentTypeID>JAL</PaymentTypeID><HostBased>True</HostBased><XMLProfile><HostBased_Manual_Settle>True</HostBased_Manual_Settle><MerchantID>3929300</MerchantID><BankID>095000</BankID><Key>3606</Key><IndustryID>R</IndustryID></XMLProfile></PaymentMethod> |

Result:

<?xml version="1.0" encoding="utf-8" ?>

```

=<Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">

  <Code>OK</Code>

  <Error>A new Merchant was successfully created!</Error>

  <Partner />

  <Vendor>315</Vendor>

  <Username>ex9</Username>

  </Result>

```

### Example 10: RS

This example table shows the information needed to add a merchant using the RS processor. The parameters should be changed when testing this example yourself.

| Parameter                | Value  |
|--------------------------|--|
| <b>UserName</b>          | Test   |
| <b>SecureToken</b>       | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>=   |
| <b>MerchantUserName</b>  | ex10   |
| <b>MerchantPassword</b>  | 123  |
| <b>AutoCloseBatch</b>    | FALSE  |
| <b>FourceDuplicate</b>   | FALSE  |
| <b>RequirePNRef</b>      | FALSE  |
| <b>ContactFirstName</b>  | John   |
| <b>ContactLastName</b>   | Doe  |
| <b>ContactEmail</b>      | JDoe@test.com  |
| <b>ContactDayPhone</b>   | 555-555-5555   |
| <b>ContactStreet1</b>    | 123 Any St   |
| <b>ContactCity</b>       | Redmond  |
| <b>ContactSate</b>       | WA   |
| <b>ContactPostalCode</b> | 98034  |
| <b>PaymentMethodXml</b>  | <PaymentMethod><ProcessorID>RS</ProcessorID><PaymentTypeID>IMAGE</PaymentTypeID><HostBased>True</HostBased><XmlProfile/></PaymentMethod> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>

= <Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">

  <Code>OK</Code>

  <Error>A new Merchant was successfully created!</Error>

  <Partner />

  <Vendor>316</Vendor>

  <Username>ex10</Username>

  </Result>
```

### Example 11: Register

This example table shows the information needed to add a merchant with a register using RS processor. The parameters should be changed when testing this example yourself.

| Parameter               | Value  |
|-------------------------|--|
| <b>UserName</b>         | Test   |
| <b>SecureToken</b>      | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>= |
| <b>MerchantUserName</b> | ex11   |
| <b>MerchantPassword</b> | 123  |
| <b>AutoCloseBatch</b>   | FALSE  |
| <b>FourceDuplicate</b>  | FALSE  |
| <b>RequirePNRef</b>     | FALSE  |
| <b>ContactFirstName</b> | John   |
| <b>ContactLastName</b>  | Doe  |
| <b>ContactEmail</b>     | JDoe@test.com  |
| <b>ContactDayPhone</b>  | 555-555-5555   |
| <b>ContactStreet1</b>   | 123 Any St   |
| <b>ContactCity</b>      | Redmond  |

|                          |   |
|--------------------------|---|
| <b>ContactSate</b>       | WA  |
| <b>ContactPostalCode</b> | 98034   |
| <b>PaymentMethodXml</b>  | <PaymentMethod><ProcessorID>RS</ProcessorID><PaymentTypeID>PAYRECEIPT</PaymentTypeID><HostBased>True</HostBased><XmlProfile /></PaymentMethod>  |
| <b>RegistersXml</b>      | <Registers><Register><RegisterName>Terminal1</RegisterName><RegisterNum>01</RegisterNum><DebitTerminalNum>01</DebitTerminalNum><EBTTerminalNum>01</EBTTerminalNum></Register></Registers> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
- <Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
  <Code>OK</Code>
  <Error>A new Merchant was successfully created!</Error>
  <Partner />
  <Vendor>317</Vendor>
  <Username>ex11</Username>
  </Result>
```

## Example 12: Heartland Payment Systems

This example table shows the information needed to add a merchant using the Heartland Payment Systems processor. The parameters should be changed when testing this example yourself.

| Parameter               | Value  |
|-------------------------|--|
| <b>UserName</b>         | Test   |
| <b>SecureToken</b>      | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>= |
| <b>MerchantUserName</b> | ex8  |
| <b>MerchantPassword</b> | 123  |
| <b>AutoCloseBatch</b>   | FALSE  |

|                          |   |
|--------------------------|---|
| <b>ForceDuplicate</b>    | FALSE   |
| <b>RequirePNRef</b>      | FALSE   |
| <b>ContactFirstName</b>  | John  |
| <b>ContactLastName</b>   | Doe   |
| <b>ContactEmail</b>      | JDoe@test.com   |
| <b>ContactDayPhone</b>   | 555-555-5555  |
| <b>ContactStreet1</b>    | 123 Any St  |
| <b>ContactCity</b>       | Redmond   |
| <b>ContactState</b>      | WA  |
| <b>ContactPostalCode</b> | 98034   |
| <b>PaymentMethodXml</b>  | <pre>&lt;PaymentMethod&gt;&lt;ProcessorID&gt;Heartland&lt;/ProcessorID&gt;&lt;PaymentTypeID&gt;DEBIT&lt;/PaymentTypeID&gt;&lt;HostBased&gt;FALSE&lt;/HostBased&gt;&lt;XmlProfile&gt;&lt;SSL2&gt;True&lt;/SSL2&gt;&lt;ConnectionMethod&gt;SSL2&lt;/ConnectionMethod&gt;&lt;Processor_ID&gt;Heartland&lt;/Processor_ID&gt;&lt;MerchantNumber&gt;888000000394&lt;/MerchantNumber&gt;&lt;TerminalNumber&gt;1515&lt;/TerminalNumber&gt;&lt;AgentBankNumber&gt;00000&lt;/AgentBankNumber&gt;&lt;ChainNumber&gt;111111&lt;/ChainNumber&gt;&lt;StoreNumber&gt;5999&lt;/StoreNumber&gt;&lt;BinNumber&gt;999995&lt;/BinNumber&gt;&lt;CustomerServicePhone&gt;123-1231234&lt;/CustomerServicePhone&gt;&lt;LocationNumber&gt;00001&lt;/LocationNumber&gt;&lt;PostalCode&gt;98052&lt;/PostalCode&gt;&lt;CategoryCode&gt;5999&lt;/CategoryCode&gt;&lt;VNumber&gt;71004021&lt;/VNumber&gt;&lt;IndustryID&gt;R&lt;/IndustryID&gt;&lt;TimezoneID&gt;PST&lt;/TimezoneID&gt;&lt;CurrencyID&gt;840&lt;/CurrencyID&gt;&lt;CountryID&gt;USA&lt;/CountryID&gt;&lt;LanguageID&gt;00&lt;/LanguageID&gt;&lt;InputDeviceID&gt;2&lt;/InputDeviceID&gt;&lt;ABANumber&gt;111111111&lt;/ABANumber&gt;&lt;SettlementAgentNumber&gt;V001&lt;/SettlementAgentNumber&gt;&lt;SharingGroup&gt;ABC&lt;/SharingGroup&gt;&lt;ReimbursementAttribute&gt;Z&lt;/ReimbursementAttribute&gt;&lt;FCSID /&gt;&lt;DebitSurcharge&gt;0&lt;/DebitSurcharge&gt;&lt;/XmlProfile&gt;&lt;/PaymentMethod&gt;</pre> |

Result:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <Result xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
```

```
<Code>OK</Code>
```

```
<Error>A new Merchant was successfully created!</Error>
```

```
<Partner />
```

```
<Vendor>314</Vendor>
```

<Username>ex12</Username>

</Result>

**Note: When using AddMerchant to programmatically add merchant accounts, there is a requirement to run a sql script that will add the Admin HOST IP. If there were no prior entries, the SQL script to run is the statement below to add the IP of Admin Host. This will create this key in AppSetting\_T table entry.**

```
INSERT INTO AppSetting_T (Application_Key, AppSetting_Key, AppSetting_Value, XmlProfile_TXT)
VALUES (8, 'AdminHostIP', 'localhost|127.0.0.1', NULL)
```

**If the key was already created by running the sql statement above, you can simple update it by running this sql statement:**

```
Update AppSetting_T set AppSetting_Value =
'localhost|127.0.0.1|WHATEVERIPYOUWANTHERE'
```

## **UpdateMerchant**

This Web service operation allows you to update existing merchant information. The URL to access this operation is: <https://localhost/admin/ws/admin.asmx?op=UpdateMerchant>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

This is a useful web service when it comes to updating business data or merchant configuration data.

| Parameter                | Value  |
|--------------------------|--|
| <b>UserName</b>          | Required. User name assigned in the payment server. This user should have administrative rights to perform this web service. |
| <b>SecureToken</b>       | Required. The secure code given by the payment server located at the Preferences menu, under Password                        |
| <b>ResellerKey</b>       | Optional. The key of the reseller under which the merchant is added  |
| <b>MerchantUsername</b>  | Required. User name for the merchant assigned in the payment server  |
| <b>MerchantPassword</b>  | Required. Password for the merchant name assigned in the payment server  |
| <b>MerchantID</b>        | Optional. ID for the merchant  |
| <b>MerchantID2</b>       | Optional. ID for a second merchant   |
| <b>AnnualSales</b>       | Optional. Annual sales of the company  |
| <b>BusinessStartDate</b> | Optional. Date the business started  |
| <b>CompanyName</b>       | Optional. Name of the company  |
| <b>DoingBusinessAs</b>   | Optional. Type of company (ex: retail)   |
| <b>Url</b>               | Optional. URL of the company website   |

|                           |   |
|---------------------------|---|
| <b>FederalTaxID</b>       | Optional. Company's federal tax ID  |
| <b>StateTaxID</b>         | Optional. Company's state tax ID  |
| <b>SalesTaxID</b>         | Optional. Company's sales tax ID  |
| <b>OwnershipType</b>      | Optional. The type of company the merchant belongs to (Ex: corporation)   |
| <b>AutoCloseBatch</b>     | Required. (True or False) Set the close batch automation  |
| <b>AutoCloseBatchHour</b> | Required with True, Optional with False. Set the close batch time   |
| <b>ForceDuplicate</b>     | Required. (True or False) Allows duplicate transaction  |
| <b>RequirePNRef</b>       | Required. (True or False)   |
| <b>ContactFirstName</b>   | Required. First name of the contact person  |
| <b>ContactLastName</b>    | Required. Last name of the contact person   |
| <b>ContactEmail</b>       | Required. Email address of the contact person   |
| <b>ContactDayPhone</b>    | Required. Day time phone number of the contact person   |
| <b>ContactFax</b>         | Optional. Fax number of the contact person  |
| <b>ContactStreet1</b>     | Required. Line one of the of the contact person's street address  |
| <b>ContactStreet2</b>     | Optional. Line two of the contact person's street address   |
| <b>ContactCity</b>        | Required. Contact person's city   |
| <b>ContactState</b>       | Required. Contact person's state  |
| <b>ContactPostalCode</b>  | Required. Contact person's zip code   |
| <b>ContactCountryCode</b> | Optional. Contact person's country  |
| <b>TimeZoneOffset</b>     | Optional. Hours off time zone   |
| <b>PaymentMethodXml</b>   | Required. Extended data in XML format, this is the XML string that will contain the configuration settings for the merchant account.  |
| <b>RegistersXml</b>       | Optional. Extended data in XML format   |
| <b>ExtData</b>            | <p>Optional. Extended data in XML format</p> <p>Note: By default, sending the email to the new user/merchant is set to false but the system will send out an email to the merchant by adding this tag: &lt;SendEmail&gt;T&lt;/SendEmail&gt;</p> <p>&lt;VirtualTerminalTipAmount&gt; T or F&lt;/VirtualTerminalTipAmount&gt;<br/>For enabling tip amount to be processed using the virtual terminal.</p> <p>&lt;VirtualTerminalTaxAmount&gt; T or F&lt;/ VirtualTerminalTaxAmount&gt; for enabling the virtual terminal tax amount to be processed using the virtual terminal.</p> <p>&lt;VirtualTerminalConvenienceAmount&gt; T or F<br/>&lt;/ VirtualTerminalConvenienceAmount&gt; for enabling the convenience amount to be processed using the virtual terminal.</p> |

|  |  |
|--|--|
|  | <p>&lt;AutoSettleMerchantEmail&gt;T or F&lt;/AutoSettleMerchantEmail&gt; for enabling the auto settlement merchant email function for the account if auto settle function is enabled on terminal based accounts.</p> |
|--|--|

**Example**

The following table contains sample data you can use to test this Web service. The parameters should be changed when testing this example yourself.

| Parameter                | Value  |
|--------------------------|--|
| <b>UserName</b>          | admin  |
| <b>SecureToken</b>       | jYizQ51MNs1wf1fcP25eB3iYRxg/u95tGvMRUhyhsIo7HIDMBhq0Lw=<br>=   |
| <b>MerchantUserName</b>  | ex8  |
| <b>MerchantPassword</b>  | 123  |
| <b>AutoCloseBatch</b>    | FALSE  |
| <b>FourceDuplicate</b>   | FALSE  |
| <b>RequirePNRef</b>      | FALSE  |
| <b>ContactFirstName</b>  | John   |
| <b>ContactLastName</b>   | Doe  |
| <b>ContactEmail</b>      | JDoe@test.com  |
| <b>ContactDayPhone</b>   | 555-555-5552   |
| <b>ContactStreet1</b>    | 123 Any St   |
| <b>ContactCity</b>       | Redmond  |
| <b>ContactSate</b>       | WA   |
| <b>ContactPostalCode</b> | 98034  |
| <b>PaymentMethodXml</b>  | <pre>&lt;PaymentMethods&gt; &lt;PaymentMethod&gt; &lt;ProcessorID&gt;Heartland&lt;/ProcessorID&gt; &lt;PaymentTypeID&gt;DEBIT&lt;/PaymentTypeID&gt; &lt;HostBased&gt;FALSE&lt;/HostBased&gt; &lt;XmlProfile&gt; &lt;SSL2&gt;True&lt;/SSL2&gt; &lt;ConnectionMethod&gt;SSL2&lt;/ConnectionMethod&gt; &lt;Processor_ID&gt;Heartland&lt;/Processor_ID&gt; &lt;MerchantNumber&gt;888000000394&lt;/MerchantNumber&gt; &lt;TerminalNumber&gt;1515&lt;/TerminalNumber&gt; &lt;AgentBankNumber&gt;000000&lt;/AgentBankNumber&gt; &lt;ChainNumber&gt;111111&lt;/ChainNumber&gt; &lt;StoreNumber&gt;5999&lt;/StoreNumber&gt; &lt;BinNumber&gt;999995&lt;/BinNumber&gt; &lt;CustomerServicePhone&gt;123-1231234&lt;/CustomerServicePhone&gt; &lt;LocationNumber&gt;00001&lt;/LocationNumber&gt; &lt;PostalCode&gt;98052&lt;/PostalCode&gt; &lt;CategoryCode&gt;5999&lt;/CategoryCode&gt; &lt;VNumber&gt;71004021&lt;/VNumber&gt; &lt;IndustryID&gt;R&lt;/IndustryID&gt; &lt;TimezoneID&gt;PST&lt;/TimezoneID&gt; &lt;CurrencyID&gt;840&lt;/CurrencyID&gt; &lt;CountryID&gt;USA&lt;/CountryID&gt; &lt;LanguageID&gt;00&lt;/LanguageID&gt; &lt;InputDeviceID&gt;2&lt;/InputDeviceID&gt; &lt;A</pre> |

|  |  |
|--|--|
|  | <pre> BANumber&gt;111111111&lt;/ABANumber&gt;&lt;SettlementAgentNumber &gt;V001&lt;/SettlementAgentNumber&gt;&lt;SharingGroup&gt;ABC&lt;/Sharing Group&gt;&lt;ReimbursementAttribute&gt;Z&lt;/ReimbursementAttribute&gt;&lt; FCSID /&gt;&lt;DebitSurcharge&gt;0&lt;/DebitSurcharge&gt;&lt;/XmlProfile&gt;&lt;/Paymen tMethod&gt;&lt;/PaymentMethods&gt; </pre> |
|--|--|

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
- <Result xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://secure.payment-engine.com/Admin/ws">
```

```
<code>OK</code>
```

```
<error>Merchant was successfully updated!</error>
```

```
<Partner>100</Partner>
```

```
<Vendor>51</Vendor>
```

```
<Username />
```

```
</Result>
```

## DeleteMerchant

This Web service operation allows you to delete a merchant. The URL to access this operation is: <https://localhost/admin/ws/admin.asmx?op=DeleteMerchant>. The text “localhost” in the URL should be replaced with the actual host name or static IP address of the payment server. Descriptions of the parameters are listed below.

| Parameter          | Value   |
|--------------------|---|
| <b>UserName</b>    | Required. Reseller name assigned in the payment server (ex: admin)  |
| <b>SecureToken</b> | Required. The Reseller’s secure code given by the payment server located at the Preferences menu, under Password  |
| <b>ResellerKey</b> | Required. The key of the reseller under which the merchant is deleted. This would be located at the Preference menu, under Password (ex: MSP/Partner Number: 100)     |
| <b>MerchantKey</b> | Required. The key of the merchant to be deleted. Located at the Manage Merchants, under Find/Edit and search for the merchant, the ID will be in the far left column. |
| <b>ExtData</b>     | Optional. Extended data in XML format   |

## Example

The following table contains sample data you can use to test this Web service. The parameters should be changed when testing this example yourself.

| Parameter          | Value  |
|--------------------|--|
| <b>UserName</b>    | admin  |
| <b>SecureToken</b> | iEeVzZQ8LeFS8o2HRXN097MZiaur9k1hEoLo1iYnLRPvN81t/xJoTA== |
| <b>ResellerKey</b> | 100  |
| <b>MerchantKey</b> | 316  |

Result:

```
<Result>
<Code>OK</Code>
<Error>Merchant 316 deleted.</Error>
<Partner />100</Partner>
<Vendor>316</Vendor>
<Username/>
</Result>
```

## AddRecurringCreditCard

This web service operation allows you to add a customer, a contract and a credit card payment method all in one call. All parameters marked as required must be supplied. Optional parameters can be left blank and the default value will be used. Default values are empty strings for *string type* and 0 for integer type. The URL to access this web service is <http://localhost/admin/ws/recurring.asmx?op=AddRecurringCreditCard>

| Parameter         | Value   |
|-------------------|---|
| <b>Username</b>   | Required. The username of the admin user.                         |
| <b>Password</b>   | Required. The password of the admin user                          |
| <b>Vendor</b>     | Required. The numerical Vendor/Merchant Key.                      |
| <b>CustomerID</b> | Required. A merchant supplied a unique identifier for a customer. |

|                        |  |
|------------------------|--|
| <b>Customer Name</b>   | Required. The customer's name is to be submitted in this field.  |
| <b>FirstName</b>       | Optional. The customer's first name.   |
| <b>LastName</b>        | Optional. The customer's last name.  |
| <b>Title</b>           | Optional. The customer's title.  |
| <b>Department</b>      | Optional. The customer' department.  |
| <b>Street1</b>         | Optional. The customer's street address 1.   |
| <b>Street2</b>         | Optional. The customer's street address 2.   |
| <b>Street3</b>         | Optional. The customer's street address3.  |
| <b>City</b>            | Optional. The customer's city.   |
| <b>StateID</b>         | Optional. The customer's 2 character State Code  |
| <b>Province</b>        | Optional. The customer's province if it is outside the USA   |
| <b>Zip</b>             | Optional. The customer's zip code if in the USA, postal code if outside the USA  |
| <b>CountryID</b>       | Optional. The customer's 3 character country code, for example, USA or CAN   |
| <b>Email</b>           | Optional. The customer's email address.  |
| <b>Mobile</b>          | Optional. The customer's mobile phone.   |
| <b>ContractID</b>      | Required. The merchant supplied unique indentifier for the contract.   |
| <b>ContractName</b>    | Optional. The contract's name.   |
| <b>BillAmt</b>         | Required. The amount to be billed in relation to the contract.   |
| <b>TaxAmt</b>          | Optional. The tax amount.  |
| <b>TotalAmt</b>        | Required. This is the total amount. $BillAmt + TaxAmt = TotalAmt$ .  |
| <b>StartDate</b>       | Required. The start date of the contract.  |
| <b>EndDate</b>         | Optional. The end date of the contract. If this date is not given, the contract will continue to run until manually cancelled or suspended by the system due to failure of payment |
| <b>BillingPeriod</b>   | Required. Specifies the Billing Period Type, used in conjunction with BillingInterval to compute the next bill date.   |
| <b>BillingInterval</b> | Required. Depending on the billing period, it can mean different things such as DAY = every X number of days,  |

|                             |  |
|-----------------------------|--|
|                             | <p>WEEK = number of times per week,</p> <p>MONTH = number fo times per month;</p> <p>YEAR = number of times per year.</p>                                    |
| <b>MaxFailures</b>          | Optional. The number of times the system will wait after each retry when a recurring payment fails to process before it puts the contract in suspended mode. |
| <b>FailureInterval</b>      | Optional. Number of days the system will wait after each payment retry when the payment fails.   |
| <b>EmailCustomer</b>        | Optional. TRUE/FALSE setting whether to email the customer regarding the status of the recurring payment.  |
| <b>EmailMerchant</b>        | Optional. TRUE/FALSE setting whether to email the merchant regarding the status of recurring payment.  |
| <b>EmailCustomerFailure</b> | Optional. TRUE/FALSE setting whether to email the customer when the recurring payment fails.   |
| <b>EmailMerchantFailure</b> | Optional. TRUE/FALSE setting whether to email the merchant when the recurring payemt fails.  |
| <b>CcAccountNum</b>         | Required. The customer's credit card number.   |
| <b>CcExpdate</b>            | Required. The credit card expiration date.   |
| <b>CcNameOnCard</b>         | Optional. The Card Holder's name as it is on the card.   |
| <b>CcStreet</b>             | Optional. The Card Holder's billing address  |
| <b>CcZip</b>                | Optional. The Card Holder's billing zip code.  |
| <b>ExtData</b>              | Optional. Extended Data.   |

## Example

| Parameter     | Value  |
|---------------|--|
| Username:     | <input type="text" value="vital"/>           |
| Password:     | <input type="text" value="1234"/>            |
| Vendor:       | <input type="text" value="1"/>               |
| CustomerID:   | <input type="text" value="44444"/>           |
| CustomerName: | <input type="text" value="Fitness Forever"/> |
| FirstName:    | <input type="text" value="John"/>            |
| LastName:     | <input type="text" value="Smith"/>           |
| Title:        | <input type="text" value="Sales Director"/>  |
| Department:   | <input type="text" value="Sales"/>           |
| Street1:      | <input type="text" value="123 Main Street"/> |
| Street2:      | <input type="text"/>                         |
| Street3:      | <input type="text"/>                         |
| City:         | <input type="text" value="Redmond"/>         |
| StateID:      | <input type="text" value="WA"/>              |

|                  |                    |
|------------------|--------------------|
| Email:           | JSmith@hotmail.com |
| DayPhone:        |                    |
| NightPhone:      |                    |
| Fax:             |                    |
| Mobile:          |                    |
| ContractID:      | 22222              |
| ContractName:    | Fitness 4ever      |
| BillAmt:         | 19.00              |
| TaxAmt:          | 1.00               |
| TotalAmt:        | 20.00              |
| StartDate:       | 02/13/07           |
| EndDate:         | 04/13/07           |
| BillingPeriod:   | DAY                |
| BillingInterval: | 2                  |
| MaxFailures:     | 1                  |
| FailureInterval: | 1                  |
| EmailCustomer:   | TRUE               |
| EmailMerchant:   | TRUE               |

|               |                  |
|---------------|------------------|
| CcAccountNum: | 5439750001500347 |
| CcExpDate:    | 1208             |
| CcNameOnCard: | Andy Chau        |
| CcStreet:     |                  |
| CcZip:        |                  |
| ExtData:      |                  |

Invoke

## Response

```
<?xml version="1.0" encoding="utf-8" ?>
=<RecurringResult xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">

  <CustomerKey>11</CustomerKey>

  <ContractKey>7</ContractKey>

  <CcInfoKey>26607</CcInfoKey>

  <CheckInfoKey />

  <code>OK</code>

  <error>OK</error>

  <Partner>100</Partner>

  <Vendor>1</Vendor>

  <Username>vital</Username>

  </RecurringResult>
```

## ***AddRecurringCheck***

This web service allows for adding a customer, a contract and a credit card payment method all in one call. All parameters marked as required must be supplied. Optional parameters can be left blank and the default value will be used. Default values are as follows, empty string for string type and 0 for integer. The URL to access this web service is: <http://localhost/admin/ws/recurring.asmx?op=AddRecurringCheck>

| Parameter         | Value   |
|-------------------|---|
| <b>Username</b>   | Required. The username of the admin user.                         |
| <b>Password</b>   | Required. The password of the admin user                          |
| <b>Vendor</b>     | Required. The numerical Vendor/Merchant Key.                      |
| <b>CustomerID</b> | Required. A merchant supplied a unique identifier for a customer. |

|                        |  |
|------------------------|--|
| <b>Customer Name</b>   | Required. The customer's name is to be submitted in this field.  |
| <b>FirstName</b>       | Optional. The customer's first name.   |
| <b>LastName</b>        | Optional. The customer's last name.  |
| <b>Title</b>           | Optional. The customer's title.  |
| <b>Department</b>      | Optional. The customer' department.  |
| <b>Street1</b>         | Optional. The customer's street address 1.   |
| <b>Street2</b>         | Optional. The customer's street address 2.   |
| <b>Street3</b>         | Optional. The customer's street address3.  |
| <b>City</b>            | Optional. The customer's city.   |
| <b>StateID</b>         | Optional. The customer's 2 character State Code  |
| <b>Province</b>        | Optional. The customer's province if it is outside the USA   |
| <b>Zip</b>             | Optional. The customer's zip code if in the USA, postal code if outside the USA  |
| <b>CountryID</b>       | Optional. The customer's 3 character country code, for example, USA or CAN   |
| <b>Email</b>           | Optional. The customer's email address.  |
| <b>Mobile</b>          | Optional. The customer's mobile phone.   |
| <b>ContractID</b>      | Required. The merchant supplied unique indentifier for the contract.   |
| <b>ContractName</b>    | Optional. The contract's name.   |
| <b>BillAmt</b>         | Optional. The amount to be billed in relation to the contract.   |
| <b>TaxAmt</b>          | Optional. The tax amount.  |
| <b>TotalAmt</b>        | Required. This is the total amount. $BillAmt + TaxAmt = TotalAmt$ .  |
| <b>StartDate</b>       | Required. The start date of the contract.  |
| <b>EndDate</b>         | Optional. The end date of the contract. If this date is not given, the contract will continue to run until manually cancelled or suspended by the system due to failure of payment |
| <b>BillingPeriod</b>   | Required. Specifies the Billing Period Type, used in conjunction with BillingInterval to compute the next bill date.   |
| <b>BillingInterval</b> | Required. Depending on the billing period, it can mean different things such as DAY = every X number of days,  |

|                             |  |
|-----------------------------|--|
|                             | <p>WEEK = number of times per week,</p> <p>MONTH = number fo times per month;</p> <p>YEAR = number of times per year.</p>                                    |
| <b>MaxFailures</b>          | Optional. The number of times the system will wait after each retry when a recurring payment fails to process before it puts the contract in suspended mode. |
| <b>FailureInterval</b>      | Optional. Number of days the system will wait after each payment retry when the payment fails.   |
| <b>EmailCustomer</b>        | Optional. TRUE/FALSE setting whether to email the customer regarding the status of the recurring payment.  |
| <b>EmailMerchant</b>        | Optional. TRUE/FALSE setting whether to email the merchant regarding the status of recurring payment.  |
| <b>EmailCustomerFailure</b> | Optional. TRUE/FALSE setting whether to email the customer when the recurring payment fails.   |
| <b>EmailMerchantFailure</b> | Optional. TRUE/FALSE setting whether to email the merchant when the recurring payemnt fails.   |
| <b>CheckType</b>            | Required. Two types of checks whether PERSONAL or BUSINESS.  |
| <b>AccountType</b>          | Required. Two types of account whether CHECKING or SAVINGS.  |
| <b>CheckNum</b>             | Optional. This is the check number.  |
| <b>MICR</b>                 | Optional. This is the scanned MICR data of the check.  |
| <b>AccountNum</b>           | Required. This is the account number.  |
| <b>TransitNum</b>           | Required. This is the transit number.  |
| <b>SS</b>                   | Optional. Social Security number of the check holder.  |
| <b>DOB</b>                  | Optional. Date of Birth of the check holder.   |
| <b>BranchCity</b>           | Optional. The city of the bank where the branch is located.  |
| <b>DL</b>                   | Optional. The driver's license number of the check holder  |
| <b>StateCode</b>            | Optional. The 2 character State Code of the driver's License of the check holder.  |
| <b>NameOnCheck</b>          | Optional. The check holder's name as it is on the check.   |
| <b>ExtData</b>              | Optional. Extended Data.   |

## Example

| Parameter     | Value  |
|---------------|--|
| Username:     | <input type="text" value="ncn1"/>            |
| Password:     | <input type="text" value="1234"/>            |
| Vendor:       | <input type="text" value="41"/>              |
| CustomerID:   | <input type="text" value="2929292"/>         |
| CustomerName: | <input type="text" value="Fitness Land"/>    |
| FirstName:    | <input type="text" value="John"/>            |
| LastName:     | <input type="text" value="Jones"/>           |
| Title:        | <input type="text" value="Sales Manager"/>   |
| Department:   | <input type="text" value="Sales"/>           |
| Street1:      | <input type="text" value="123 Main Street"/> |
| Street2:      | <input type="text"/>                         |
| Street3:      | <input type="text"/>                         |
| City:         | <input type="text" value="Atlanta"/>         |
| StateID:      | <input type="text" value="GA"/>              |
| Province:     | <input type="text"/>                         |

|                |                  |
|----------------|------------------|
| PostalCode:    | 41344            |
| CountryID:     | USA              |
| Email:         | john@hotmail.com |
| DayPhone:      | 912-333-7777     |
| NightPhone:    |                  |
| Fax:           |                  |
| Mobile:        | 912-888-7799     |
| ContractID:    | 898989           |
| ContractName:  | Fit4now          |
| BillAmt:       | 25.00            |
| TaxAmt:        | 0.00             |
| TotalAmt:      | 25.00            |
| StartDate:     | 04/20/07         |
| EndDate:       | 04/20/08         |
| BillingPeriod: | MONTH            |

|                       |  |
|-----------------------|--|
| BillingInterval:      | <input type="text" value="1"/>                     |
| MaxFailures:          | <input type="text" value="2"/>                     |
| FailureInterval:      | <input type="text" value="2"/>                     |
| EmailCustomer:        | <input type="text" value="TRUE"/>                  |
| EmailMerchant:        | <input type="text" value="TRUE"/>                  |
| EmailCustomerFailure: | <input type="text" value="TRUE"/>                  |
| EmailMerchantFailure: | <input type="text" value="TRUE"/>                  |
| CheckType:            | <input type="text" value="BUSINESS"/>              |
| AccountType:          | <input type="text" value="CHECKING"/>              |
| CheckNum:             | <input type="text" value="1001"/>                  |
| MICR:                 | <input type="text" value="90000018 100124413815"/> |
| AccountNum:           | <input type="text" value="24413815"/>              |
| TransitNum:           | <input type="text" value="90000018"/>              |
| SS:                   | <input type="text"/>                               |
| DOB:                  | <input type="text"/>                               |
| BranchCity:           | <input type="text"/>                               |
| DL:                   | <input type="text"/>                               |
| StateCode:            | <input type="text"/>                               |
| NameOnCheck:          | <input type="text" value="John Jones"/>            |
| ExtData:              | <input type="text"/>                               |

## Response

<?xml version="1.0" encoding="utf-8" ?>

```

- <RecurringResult xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">

  <CustomerKey>12</CustomerKey>

  <ContractKey>8</ContractKey>

  <CcInfoKey />

  <CheckInfoKey>3</CheckInfoKey>

  <code>OK</code>

  <error>OK</error>

  <Partner>100</Partner>

  <Vendor>41</Vendor>

  <Username>ncn1</Username>

  </RecurringResult>

```

## ***ProcessCreditCard - Recurring Billing***

This web service operation processes credit card transactions within the recurring billing module. The URL to access this web service is:

<http://localhost/admin/ws/recurring.asmx?op=ProcessCreditCard>

| Parameter        | Value   |
|------------------|---|
| <b>Username</b>  | Required. The username of the admin user.                         |
| <b>Password</b>  | Required. The password of the admin user                          |
| <b>Vendor</b>    | Required. The numerical Vendor/Merchant Key.                      |
| <b>CcInfoKey</b> | Required. The numerical Credit Card Info key.                     |
| <b>Amount</b>    | Required. The amount that will be processed for that transaction. |
| <b>InvNum</b>    | Optional. The associated invoice number.                          |
| <b>ExtData</b>   | Optional. Extended Data.  |

## Example

| Parameter  | Value                              |
|------------|------------------------------------|
| Username:  | <input type="text" value="vital"/> |
| Password:  | <input type="text" value="1234"/>  |
| Vendor:    | <input type="text" value="1"/>     |
| CcInfoKey: | <input type="text" value="26607"/> |
| Amount:    | <input type="text" value="1.00"/>  |
| InvNum:    | <input type="text" value="12345"/> |
| ExtData:   | <input type="text"/>               |

## Response:

```
<?xml version="1.0" encoding="utf-8" ?>  
  
= <RecurringResult xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
  xmlns="http://secure.payment-engine.com/Admin/ws">  
  
  <code>OK</code>  
  
  <error>APPROVED</error>  
  
  <Result>0</Result>  
  
  <AuthCode>TAS351</AuthCode>  
  
  <PNRef>26873</PNRef>  
  
  <Message>NO MATCH</Message>  
  
  </RecurringResult>
```

## ProcessCheck – Recurring Billing

This web service operation allows for the processing of check transactions within the recurring billing module. The URL to access this Web Service is:

<http://localhost/admin/ws/recurring.asmx?op=ProcessCheck>

| Parameter           | Value   |
|---------------------|---|
| <b>Username</b>     | Required. The username of the admin user.                         |
| <b>Password</b>     | Required. The password of the admin user                          |
| <b>Vendor</b>       | Required. The numerical Vendor/Merchant Key.                      |
| <b>CheckInfoKey</b> | Required. The numerical Check Payment Info key.                   |
| <b>Amount</b>       | Required. The amount that will be processed for that transaction. |
| <b>InvNum</b>       | Optional. The associated invoice number.                          |
| <b>ExtData</b>      | Optional. Extended Data.  |

### Example

| Parameter     | Value                              |
|---------------|------------------------------------|
| Username:     | <input type="text" value="ncn1"/>  |
| Password:     | <input type="text" value="1234"/>  |
| Vendor:       | <input type="text" value="41"/>    |
| CheckInfoKey: | <input type="text" value="7"/>     |
| Amount:       | <input type="text" value="5.00"/>  |
| InvNum:       | <input type="text" value="12345"/> |
| ExtData:      | <input type="text"/>               |

### Response

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <RecurringResult xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
```

```
<code>OK</code>
```

<error>**APPROVED**</error>

<Result>**0**</Result>

<AuthCode>**AUTH NUM 674-564**</AuthCode>

<PNRef>**26945**</PNRef>

<Message>**APPROVAL**</Message>

</RecurringResult>

## ***ManageCheckInfo***

This Web Service operation allows for managing check information .The URL to access this Web Service is: <http://localhost/admin/ws/recurring.asmx?op=ManageCheckInfo>

| Parameter           | Value   |
|---------------------|---|
| <b>Username</b>     | Required. The username of the admin user.   |
| <b>Password</b>     | Required. The password of the admin user  |
| <b>Vendor</b>       | Required. The numerical Vendor/Merchant Key.  |
| <b>CustomerKey</b>  | Required. The numerical customer key.   |
| <b>CheckInfoKey</b> | Required for TransType UPDATE and DELETE. The numerical Customer Key                            |
| <b>CheckType</b>    | Required. Two types of checks whether PERSONAL or BUSINESS.                                     |
| <b>AccountType</b>  | Required. Two types of account whether CHECKING or SAVINGS.                                     |
| <b>CheckNum</b>     | Optional. This is the check number.   |
| <b>MICR</b>         | Optional. This is the scanned MICR data of the check.   |
| <b>AccountNum</b>   | Required. This is the account number.   |
| <b>TransitNum</b>   | Required. This is the transit number.   |
| <b>SS</b>           | Optional. Social Security number of the check holder.   |
| <b>DOB</b>          | Optional. Date of Birth of the check holder.  |
| <b>BranchCity</b>   | Optional. The city of the bank where the branch is located.                                     |
| <b>DL</b>           | Optional. The driver's license number of the check holder                                       |
| <b>StateCode</b>    | Optional. The 2 character State Code of the driver's license of the check holder, e.g. NY or GA |
| <b>NameOnCheck</b>  | Optional. The check holder's name as it is on the check.  |
| <b>Email</b>        | Optional. The customer's email address.   |

|                   |  |
|-------------------|--|
| <b>DayPhone</b>   | Optional. The customer's day phone.  |
| <b>Street1</b>    | Optional. The customer's street address 1.                                 |
| <b>Street2</b>    | Optional. The customer's street address 2.                                 |
| <b>Street3</b>    | Optional. The customer's street address3.                                  |
| <b>City</b>       | Optional. The customer's city.   |
| <b>StateID</b>    | Optional. The customer's 2 character State Code                            |
| <b>Province</b>   | Optional. The customer's province if it is outside the USA                 |
| <b>PostalCode</b> | Optional. The customer's zip code if in USA , postal code if outside USA   |
| <b>CountryID</b>  | Optional. The customer's 3 character country code, for example, USA or CAN |
| <b>ExtData</b>    | Optional. Extended Data.   |

## Example

| Parameter     | Value  |
|---------------|--|
| Username:     | <input type="text" value="ncn1"/>                  |
| Password:     | <input type="text" value="1234"/>                  |
| TransType:    | <input type="text" value="ADD"/>                   |
| Vendor:       | <input type="text" value="41"/>                    |
| CustomerKey:  | <input type="text" value="12"/>                    |
| CheckInfoKey: | <input type="text" value="3"/>                     |
| CheckType:    | <input type="text" value="PERSONAL"/>              |
| AccountType:  | <input type="text" value="CHECKING"/>              |
| CheckNum:     | <input type="text" value="1001"/>                  |
| MICR:         | <input type="text" value="490000018100124413815"/> |
| AccountNum:   | <input type="text" value="24413815"/>              |
| TransitNum:   | <input type="text" value="490000018"/>             |
| RawMICR:      | <input type="text"/>                               |
| SS:           | <input type="text"/>                               |

|              |  |
|--------------|--|
| DOB:         | <input type="text"/>                         |
| BranchCity:  | <input type="text"/>                         |
| DL:          | <input type="text"/>                         |
| StateCode:   | <input type="text"/>                         |
| NameOnCheck: | <input type="text"/>                         |
| Email:       | <input type="text"/>                         |
| DayPhone:    | <input type="text"/>                         |
| Street1:     | <input type="text" value="123 Main Street"/> |
| Street2:     | <input type="text"/>                         |
| Street3:     | <input type="text"/>                         |
| City:        | <input type="text" value="Atlanta"/>         |
| StateID:     | <input type="text" value="GA"/>              |
| Province:    | <input type="text"/>                         |
| PostalCode:  | <input type="text" value="40190"/>           |
| CountryID:   | <input type="text" value="USA"/>             |
| ExtData:     | <input type="text"/>                         |

## Response

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <RecurringResult xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
```

```
<CustomerKey>12</CustomerKey>
```

```
<ContractKey />
```

```
<CcInfoKey />
```

<CheckInfoKey>4</CheckInfoKey>

<code>OK</code>

<error>OK</error>

<Partner>100</Partner>

<Vendor>41</Vendor>

<Username>ncn1</Username>

</RecurringResult>

## ***ManageCreditCardInfo***

This Web Service allows for managing the credit card information. The URL for accessing this Web Service is at:

<http://localhost/admin/ws/recurring.asmx?op=ManageCreditCardInfo>

| Parameter           | Value   |
|---------------------|---|
| <b>Username</b>     | Required. The username of the admin user.   |
| <b>Password</b>     | Required. The password of the admin user  |
| <b>TransType</b>    | Required. The type of transaction being performed. Valid values are:<br>ADD<br>UPDATE<br>DELETE |
| <b>Vendor</b>       | Required. The numerical Vendor/Merchant Key.  |
| <b>CustomerKey</b>  | Required. The numerical Customer Key.   |
| <b>CardInfoKey</b>  | Required. The numerical credit card info key.   |
| <b>CcAccountNum</b> | Required. The credit card account number.   |
| <b>CcExpDate</b>    | Required. The credit card expiration date.  |
| <b>CcNameonCard</b> | Optional. The name of the card holder   |
| <b>CcStreet</b>     | Optional. The card holder's billing address.  |
| <b>CcZip</b>        | Optional. The card holder's billing zip code.   |
| <b>ExtData</b>      | Optional. Extended Data.  |

## Example

| Parameter     | Value   |
|---------------|---|
| Username:     | <input type="text" value="vital"/>            |
| Password:     | <input type="text" value="1234"/>             |
| TransType:    | <input type="text" value="ADD"/>              |
| Vendor:       | <input type="text" value="1"/>                |
| CustomerKey:  | <input type="text" value="11"/>               |
| CardInfoKey:  | <input type="text" value="26607"/>            |
| CcAccountNum: | <input type="text" value="5439750001500347"/> |
| CcExpDate:    | <input type="text" value="1209"/>             |
| CcNameOnCard: | <input type="text" value="John Smith"/>       |
| CcStreet:     | <input type="text" value="123 Main Street"/>  |
| CcZip:        | <input type="text" value="40490"/>            |
| ExtData:      | <input type="text"/>                          |

## Response

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<RecurringResult xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
```

```
<CustomerKey>11</CustomerKey>
```

```
<ContractKey />
```

```
<CcInfoKey>26674</CcInfoKey>
```

```
<CheckInfoKey />
```

```
<code>OK</code>
```

```
<error>OK</error>
```

<Partner>100</Partner>

<Vendor>1</Vendor>

<Username>vital</Username>

</RecurringResult>

## **ManageContract**

This web service allows for managing different properties of contracts via integration. This can be accessed by using this URL:

<http://localhost/admin/ws/recurring.asmx?op=ManageContract>

| Parameter             | Value   |
|-----------------------|---|
| <b>Username</b>       | Required. The username of the admin user.   |
| <b>Password</b>       | Required. The password of the admin user  |
| <b>TransType</b>      | Required. The type of transaction being performed. The valid values are:<br>ADD<br>UPDATE<br>DELETE   |
| <b>Vendor</b>         | Required. The numerical Vendor/Merchant Key.  |
| <b>CustomerKey</b>    | Required. The numerical customer key.   |
| <b>ContractKey</b>    | Required for TransType UPDATE and DELETE. The numerical contract key.   |
| <b>PaymentInfoKey</b> | Required for Transtype UPDATE and ADD. The numerical information Key. This is dependent for the PaymentType. If you set the Payment Type to CC then the information that needs to be passed in this field is the CCInfoKey or CardInfoKey. Now if CK was set at the PaymentType, then the information that needs to go in this field is the CheckInfoKey. Please make sure that you are passing the right key based on the PaymentType. |
| <b>PaymentType</b>    | Required for TransType ADD and UPDATE. Type of payment:<br>CC for Credit Card and CK for Check  |
| <b>CustomerID</b>     | Required. A merchant supplied a unique identifier for a customer.   |
| <b>Customer Name</b>  | Required. The customer's name is to be submitted in this field.   |
| <b>FirstName</b>      | Optional. The customer's first name.  |
| <b>LastName</b>       | Optional. The customer's last name.   |
| <b>Title</b>          | Optional. The customer's title.   |
| <b>Department</b>     | Optional. The customer' department.   |

|                        |  |
|------------------------|--|
| <b>Street1</b>         | Optional. The customer's street address 1.   |
| <b>Street2</b>         | Optional. The customer's street address 2.   |
| <b>Street3</b>         | Optional. The customer's street address3.  |
| <b>City</b>            | Optional. The customer's city.   |
| <b>StateID</b>         | Optional. The customer's 2 character State Code  |
| <b>Province</b>        | Optional. The customer's province if it is outside the USA   |
| <b>Zip</b>             | Optional. The customer's zip code if in the USA, postal code if outside the USA  |
| <b>CountryID</b>       | Optional. The customer's 3 character country code, for example, USA or CAN   |
| <b>DayPhone</b>        | Optional. The customer's day phone.  |
| <b>NightPhone</b>      | Optional. The customer's evening phone.  |
| <b>Fax</b>             | Optional. The customer's fax number.   |
| <b>Email</b>           | Optional. The customer's email address.  |
| <b>Mobile</b>          | Optional. The customer's mobile phone.   |
| <b>ContractID</b>      | Required. The merchant supplied unique identifier for the contract.  |
| <b>ContractName</b>    | Optional. The contract's name.   |
| <b>BillAmt</b>         | Optional. The amount to be billed in relation to the contract.   |
| <b>TaxAmt</b>          | Optional. The tax amount.  |
| <b>TotalAmt</b>        | Required. This is the total amount. BillAmt + TaxAmt = TotalAmt.   |
| <b>StartDate</b>       | Required. The start date of the contract.  |
| <b>EndDate</b>         | Optional. The end date of the contract. If this date is not given, the contract will continue to run until manually cancelled or suspended by the system due to failure of payment |
| <b>NextBillDt</b>      | Required. This is required for TRANSTYPE ADD and UPDATE.   |
| <b>BillingPeriod</b>   | Required. Specifies the Billing Period Type, used in conjunction with BillingInterval to compute the next bill date.   |
| <b>BillingInterval</b> | Required. Depending on the billing period, it can mean different things such   |

|                             |   |
|-----------------------------|---|
|                             | <p>as DAY = every X number of days,</p> <p>WEEK = number of times per week,</p> <p>MONTH = number fo times per month;</p> <p>YEAR = number of times per year.</p> |
| <b>MaxFailures</b>          | Optional. The number of times the system will wait after each retry when a recurring payment fails to process before it puts the contract in suspended mode.      |
| <b>FailureInterval</b>      | Optional. Number of days the system will wait after each payment retry when the payment fails.  |
| <b>EmailCustomer</b>        | Optional. TRUE/FALSE setting whether to email the customer regarding the status of the recurring payment.   |
| <b>EmailMerchant</b>        | Optional. TRUE/FALSE setting whether to email the merchant regarding the status of recurring payment.   |
| <b>EmailCustomerFailure</b> | Optional. TRUE/FALSE setting whether to email the customer when the recurring payment fails.  |
| <b>EmailMerchantFailure</b> | Optional. TRUE/FALSE setting whether to email the merchant when the recurring payemnt fails.  |
| <b>Status</b>               | Optional. Status of the contract.   |
| <b>ExtData</b>              | Optional. Extended Data.  |
|                             |   |

### Example

| Parameter       | Value                               |
|-----------------|-------------------------------------|
| Username:       | <input type="text" value="vital"/>  |
| Password:       | <input type="text" value="1234"/>   |
| TransType:      | <input type="text" value="UPDATE"/> |
| Vendor:         | <input type="text" value="1"/>      |
| CustomerKey:    | <input type="text" value="11"/>     |
| ContractKey:    | <input type="text" value="7"/>      |
| PaymentInfoKey: | <input type="text" value="26607"/>  |

|               |                 |
|---------------|-----------------|
| PaymentType:  | CC              |
| CustomerID:   | 44444           |
| CustomerName: | Fitness Forever |
| FirstName:    | John            |
| LastName:     | Smith           |
| Title:        | Sales Manager   |
| Department:   | Sales           |
| Street1:      | 123 Main Street |
| Street2:      |                 |

|               |                   |
|---------------|-------------------|
| Street3:      |                   |
| City:         | Atlanta           |
| StateID:      | GA                |
| Province:     |                   |
| Zip:          | 40490             |
| CountryID:    | USA               |
| Email:        | johns@hotmail.com |
| DayPhone:     | 912-333-4444      |
| NightPhone:   | 912-888-9999      |
| Fax:          |                   |
| Mobile:       | 912-777-9999      |
| ContractID:   | 22222             |
| ContractName: | Fitness4ever      |
| BillAmt:      | 19.00             |
| TaxAmt:       | 2.00              |
| TotalAmt:     | 21.00             |
| StartDate:    | 02/13/2007        |
| EndDate:      | 04/13/2007        |

|                       |  |
|-----------------------|--|
| NextBillDt:           | <input type="text" value="2/13/2007"/> |
| BillingPeriod:        | <input type="text" value="DAY"/>       |
| BillingInterval:      | <input type="text" value="2"/>         |
| MaxFailures:          | <input type="text" value="1"/>         |
| FailureInterval:      | <input type="text" value="1"/>         |
| EmailCustomer:        | <input type="text" value="TRUE"/>      |
| EmailMerchant:        | <input type="text" value="TRUE"/>      |
| EmailCustomerFailure: | <input type="text" value="TRUE"/>      |
| EmailMerchantFailure: | <input type="text" value="TRUE"/>      |
| Status:               | <input type="text"/>                   |
| ExtData:              | <input type="text"/>                   |

<?xml version="1.0" encoding="utf-8" ?>

= <RecurringResult xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://secure.payment-engine.com/Admin/ws">

<CustomerKey>**11**</CustomerKey>

<ContractKey>**7**</ContractKey>

<CcInfoKey>**26607**</CcInfoKey>

<CheckInfoKey />

<code>**OK**</code>

<error>**OK**</error>

<Partner>**100**</Partner>

<Vendor>**1**</Vendor>

<Username>**vital**</Username>

</RecurringResult>

## ***ManageCustomer***

This webservice allows for the management of customer information. This web service can be accessed by this url:

<http://savqaps/admin/ws/recurring.asmx?op=ManageCustomer>

| Parameter            | Value   |
|----------------------|---|
| <b>Username</b>      | Required. The username of the admin user.   |
| <b>Password</b>      | Required. The password of the admin user  |
| <b>TransType</b>     | Required. The type of transaction being performed. The valid values are:<br>ADD<br>UPDATE<br>DELETE |
| <b>Vendor</b>        | Required. The numerical Vendor/Merchant Key.  |
| <b>CustomerKey</b>   | Required. The numerical customer key.   |
| <b>CustomerID</b>    | Required. A merchant supplied a unique identifier for a customer.                                   |
| <b>Customer Name</b> | Required. The customer's name is to be submitted in this field.                                     |
| <b>FirstName</b>     | Optional. The customer's first name.  |
| <b>LastName</b>      | Optional. The customer's last name.   |
| <b>Title</b>         | Optional. The customer's title.   |
| <b>Department</b>    | Optional. The customer's department.  |
| <b>Street1</b>       | Optional. The customer's street address 1.  |
| <b>Street2</b>       | Optional. The customer's street address 2.  |
| <b>Street3</b>       | Optional. The customer's street address3.   |
| <b>City</b>          | Optional. The customer's city.  |
| <b>StateID</b>       | Optional. The customer's 2 character State Code   |
| <b>Province</b>      | Optional. The customer's province if it is outside the USA  |
| <b>Zip</b>           | Optional. The customer's zip code if in the USA, postal code if outside the USA                     |
| <b>CountryID</b>     | Optional. The customer's 3 character country code, for example, USA or CAN                          |
| <b>DayPhone</b>      | Optional. The customer's day phone.   |
| <b>NightPhone</b>    | Optional. The customer's evening phone.   |

|                |   |
|----------------|---|
| <b>Fax</b>     | Optional. The customer's fax number.    |
| <b>Email</b>   | Optional. The customer's email address. |
| <b>Mobile</b>  | Optional. The customer's mobile phone.  |
| <b>Status</b>  | Optional. Status of the contract.       |
| <b>ExtData</b> | Optional. Extended Data.                |
|                |   |

### Example

| Parameter     | Value  |
|---------------|--|
| Username:     | <input type="text" value="vital"/>           |
| Password:     | <input type="text" value="1234"/>            |
| TransType:    | <input type="text" value="UPDATE"/>          |
| Vendor:       | <input type="text" value="1"/>               |
| CustomerKey:  | <input type="text" value="11"/>              |
| CustomerID:   | <input type="text" value="44444"/>           |
| CustomerName: | <input type="text" value="Fitness Forever"/> |
| FirstName:    | <input type="text" value="John"/>            |
| LastName:     | <input type="text" value="Smith"/>           |
| Title:        | <input type="text" value="Sales Manager"/>   |
| Department:   | <input type="text" value="Sales"/>           |
| Street1:      | <input type="text" value="123 Main Street"/> |
| Street2:      | <input type="text"/>                         |
| Street3:      | <input type="text"/>                         |
| City:         | <input type="text" value="Atlanta"/>         |
| StateID:      | <input type="text" value="GA"/>              |
| Province:     | <input type="text"/>                         |

|                                       |   |
|---------------------------------------|---|
| Zip:                                  | <input type="text" value="40490"/>              |
| CountryID:                            | <input type="text" value="USA"/>                |
| Email:                                | <input type="text" value="JSmith@hotmail.com"/> |
| DayPhone:                             | <input type="text" value="912-333-4444"/>       |
| NightPhone:                           | <input type="text" value="912-888-9999"/>       |
| Fax:                                  | <input type="text"/>                            |
| Mobile:                               | <input type="text" value="912-777-8888"/>       |
| Status:                               | <input type="text"/>                            |
| ExtData:                              | <input type="text"/>                            |
| <input type="button" value="Invoke"/> |   |

## Response

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <RecurringResult xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
```

```
<CustomerKey>11</CustomerKey>
```

```
<code>OK</code>
```

```
<error>OK</error>
```

```
<Partner />
```

```
<Vendor>1</Vendor>
```

```
<Username>vital</Username>
```

```
</RecurringResult>
```

## ManageContractAddDaysToNextBillDt

This web service allows for adding days to the next billing date and it can be accessed by this URL:

<http://localhost/admin/ws/recurring.asmx?op=ManageContractAddDaysToNextBillDt>

| Parameter          | Value   |
|--------------------|---|
| <b>Username</b>    | Required. The username of the admin user.                             |
| <b>Password</b>    | Required. The password of the admin user                              |
| <b>Vendor</b>      | Required. The numerical Vendor/Merchant Key.                          |
| <b>CustomerKey</b> | Required. The numerical customer key.                                 |
| <b>ContractKey</b> | Required for TransType UPDATE and DELETE. The numerical contract key. |
| <b>NumOfDays</b>   | The number of days to be added.                                       |
| <b>ExtData</b>     | Optional. Extended Data.  |

### Example

| Parameter    | Value                              |
|--------------|------------------------------------|
| Username:    | <input type="text" value="vital"/> |
| Password:    | <input type="text" value="1234"/>  |
| Vendor:      | <input type="text" value="1"/>     |
| CustomerKey: | <input type="text" value="11"/>    |
| ContractKey: | <input type="text" value="7"/>     |
| NumOfDays:   | <input type="text" value="15"/>    |
| ExtData:     | <input type="text"/>               |

### Response

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
= <RecurringResult xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://secure.payment-engine.com/Admin/ws">
```

```
<CustomerKey>11</CustomerKey>
```

```
<ContractKey>7</ContractKey>
<CcInfoKey />
<CheckInfoKey />
<code>OK</code>
<error>NextBillDate=2/28/2007</error>
<Partner />
<Vendor />
<Username>vital</Username>
</RecurringResult>
```

## GetNetworkID

This web service allows for returning the debit network ID if the debit card number matches any of these network's bin ranges. If there is a match, the card can likely be used as a debit card and processed through the Debit Network. This BIN range is stored in %EPSROOT%\xmlfiles\CardBin.txt which needs to be periodically updated MANUALLY probably every month or so.

This web service can be accessed through this URL:  
<http://localhost/smartpayments/validate.asmx?op=GetNetworkID>

Note: This was tested with a live Debit card number.

## Example

| Parameter         | Value                                       |
|-------------------|---|
| <b>Username</b>   | Required. The username of the admin user.   |
| <b>Password</b>   | Required. The password of the admin user    |
| <b>CardNumber</b> | Required. The customer's debit card number. |

Sample: <http://secure.payment-engine.com/smartpayments/validate.asmx?op=GetNetworkID>

```
POST /smartpayments/validate.asmx/GetNetworkID HTTP/1.1
Host: secure.payment-engine.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length
```

**UserName=string&Password=string&CardNumber=string**

| Parameter   | Value   |
|-------------|---|
| UserName:   | <input type="text" value="test"/>             |
| Password:   | <input type="text" value="123"/>              |
| CardNumber: | <input type="text" value="XXXXXXXXXXXX2557"/> |

### Response

```
<?xml version="1.0" encoding="utf-8" ?>
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://localhost/SmartPayments/">
  <Result>0</Result>
  <RespMSG>Interlink</RespMSG>
  <Message>Y</Message>
  <Message1>NetworkID=ILK,NetworkAuthorizerNum=48</Message1>
  <Message2>OfflineSupported=True</Message2>
  <AuthCode>ILK</AuthCode>
  <HostCode>48</HostCode>
</Response>
```

## BIN Management Network Table Values

| Debit Network        | Network ID | Network Authorization Number |
|----------------------|------------|------------------------------|
| Accel                | ACL        | 69                           |
| AFFN                 | AFN        | 68                           |
| Alaska Option        | AKO        | 61                           |
| CU24                 | C24        | 85                           |
| Interlink            | ILK        | 48                           |
| Jeanie               | JEN        | 86                           |
| Star Northeast (MAC) | MAC        | 17                           |
| Maestro              | MAE        | 40                           |
| Nets                 | NET        | 83                           |
| NYCE                 | NYC        | 28                           |
| Pulse                | PUL        | 06                           |
| Star Southeast       | SES        | 07                           |
| Shazam               | SHZ        | 58                           |
| Star West            | STX        | 23                           |
| TYME                 | TYM        | 78                           |

## Web Service Response Field

### *Transact.asmx*

| Response Field  | Data Type                           | Value Description  | Remarks  |
|-----------------|-------------------------------------|--|--|
| <b>AuthCode</b> | A string value up to 50 characters  | Returns the transaction result code from the payment processor | This value can be either an approval code, for approved transactions, or an error code, for declined transactions  |
| <b>ExtData</b>  | A string value up to 500 characters | Returns extra data from the processed transaction              | The value of ExtData will be in a specific format. The format typically consists of the name of the data field, an equal sign, and then the value for the data field. Multiple data fields are separated with a comma. See the "Web Service ExtData Response Field Data Elements" for full description of data elements that can be returned. The following is an example of the |

|                          |   |  |  |
|--------------------------|---|--|--|
|                          |   |  | format:<br><i>ExtName1=ExtValue1,ExtName2=ExtValue2</i>  |
| <b>GetAVSResult</b>      | A string value up to 1 character            | Returns the overall address verification result code from the payment processor                    | When programmatically validating an AVS Result, this value should ALWAYS be used instead of any formatted response message describing the result   |
| <b>GetAVSResultTXT</b>   | A string value up to 25 characters          | Returns the formatted response message when address verification is performed                      | Do NOT use this when programmatically validating a transaction's AVS result; please see GetAVSResult field   |
| <b>GetCommercialCard</b> | A string value representing a Boolean value | Returns the payment processor's response indicator that specifies if the card is a commercial card | This value is only applicable to credit card transactions. The card verification number is typically printed on the back of the card and not embossed on the front. It is used as an extra authentication method for "card not present" transactions. When programmatically validating a CV Result, this value should ALWAYS be used instead of any formatted response message describing the result |
| <b>GetCVResult</b>       | A string value up to 1 character            | Returns the card verification result code from the payment processor                               | This value is only applicable to credit card transactions. The card verification number is typically printed on the back of the card and not embossed on the front. It is used as an extra authentication method for "card not present" transactions. When programmatically validating a CV Result, this value should ALWAYS be used instead of any formatted response message describing the result |
| <b>GetCVResultTXT</b>    | A string value up to 25 characters          | Returns the formatted response message when card verification is performed                         | This value is only applicable to credit card transactions. Do NOT use this when programmatically validating a transaction's CV result; please see GetCVResult field  |
| <b>GetStreetMatchTXT</b> | A string value up to 25 characters          | Returns the formatted response message when street number address verification is performed        | This value will typically be "Match", for correctly matching the street address, or "No Match", for an incorrect street address  |
| <b>GetZipMatchTXT</b>    | A string value up to 25 characters          | Returns the formatted response message when zip code address verification is performed             | This value will typically be "Match", for correctly matching the zip code, or "No Match", for an incorrect zip code  |
| <b>HostCode</b>          | A string value up to 30 characters          | Typically returns a number which uniquely identifies the transaction in the payment processor      | This value may not be returned for all payment processors  |
| <b>Message</b>           | A string value up to 50 characters          | Returns a formatted response message concerning the processed transaction                          | This value will typically be "APPROVAL", for approved transactions, or an error message, for declined transactions. Do NOT use this when programmatically validating a transaction's result; please see Result field below   |
| <b>Message1</b>          | A string value up to 50 characters          | Returns an extra formatted response message giving more information about the                      | The Payment Server will only populate this field when there is applicable information from the payment processor to return   |

|                 |   |   |   |
|-----------------|---|---|---|
|                 |   | processed transaction   |   |
| <b>Message2</b> | A string value up to 50 characters                  | Returns an extra formatted response message giving more information about the processed transaction                                       | The Payment Server will only populate this field when there is applicable information from the payment processor to return  |
| <b>PNRef</b>    | A string value representing a signed 32-bit integer | Returns a number which uniquely identifies the transaction in the payment gateway   |   |
| <b>RespMSG</b>  | A string value up to 50 characters                  | Returns the response message concerning the processed transaction   | This value is typically either Approved or Declined. Do NOT use this when programmatically validating a transaction's result; please see Result field below   |
| <b>Result</b>   | A string value representing a signed 32-bit integer | Returns the transaction result code from the payment gateway which signifies the result of the transaction (i.e. approved, decline, etc.) | When programmatically validating a transaction's result, this value should ALWAYS be used instead of any response message describing the result. See the "Result Response Fields Definitions" section for a full list of result values and descriptions |

### ***Transact.asmx Web Service ExtData Response Field Data Elements***

| Data Element Name | Value Description  | Remarks  |
|-------------------|--|--|
| <b>BatchNum</b>   | Returns the current batch number, returned by the payment processor, for transactions, settlement, and batch inquiries | Not all payment processors support returning this data element |
| <b>CardType</b>   | Returns the credit card type (VISA, MASTERCARD, etc), payment method (Debit, EBT, or EGC) for card-based payments      | This value is not returned for Check/ACH payments              |
| <b>InvNum</b>     | Returns the same invoice number for the transaction that was originally sent in the request to the Payment Server      |  |

### ***TrxDetail.asmx***

| Response Field         | Data Type                           | Value Description  | Remarks  |
|------------------------|-------------------------------------|--|--|
| <b>Account_Type_CH</b> | A string value up to 10 characters  | Returns the card type of the transaction, e.g. VISA, Diners, EBT |  |
| <b>AccountNum_VC</b>   | A string value up to 200 characters | Returns the check account number                                 | This field will be masked out with asterisk (*) characters except for the last 4 |

|                          |   |   |  |
|--------------------------|---|---|--|
|                          |   |   | digits if the System Security Level of the user is set to 1  |
| <b>Acct_Num_CH</b>       | A string value up to 200 characters                     | Returns credit card number  | This field will be masked out with asterisk (*) characters except for the last 4 digits if the System Security Level of the user is set to 1 |
| <b>Amount_MN</b>         | A string value representing a signed 64-bit real number | Returns the check's total amount  |  |
| <b>Approval_Code_CH</b>  | A string value up to 50 characters                      | Returns the response code from the payment processor                                    |  |
| <b>Auth_Amt_MN</b>       | A string value representing a signed 64-bit real number | Returns the authorized amount of a card transaction                                     |  |
| <b>Authorization</b>     | A string value representing a signed 64-bit real number | Returns the dollar amount of all Authorization (PreAuth) transactions                   |  |
| <b>Authorization_Cnt</b> | A string value representing a signed 32-bit integer     | Returns the transaction count of all Authorization (PreAuth) transactions               |  |
| <b>AVS_Resp_CH</b>       | A string value up to 1 character                        | Returns the address verification result code from the payment processor                 |  |
| <b>AVS_Resp_Txt_VC</b>   | A string value up to 25 characters                      | Returns the formatted response message when address verification is performed           |  |
| <b>Batch_Number</b>      | A string value up to 10 characters                      | Returns the batch number for the transaction that was returned by the payment processor | Not all payment processors support returning this data element   |
| <b>Capture</b>           | A string value representing a signed 64-bit real number | Returns the dollar amount of all Capture transactions                                   | This value will always return "0"  |
| <b>Capture_Cnt</b>       | A string value representing a signed 32-bit integer     | Returns the transaction count of all Capture transactions                               | This value will always return "0"  |
| <b>Card_Info_Key</b>     | A string value representing a signed 32-bit integer     | Returns the primary key of the CC_Info_T table in the database                          |  |
| <b>Cash_Back_Amt_MN</b>  | A string value representing a signed                    | Returns the cash back amount for a debit or   |  |

|                         |   |  |  |
|-------------------------|---|--|--|
|                         | 64-bit real number                                      | EBT transaction  |  |
| <b>CheckNum_CH</b>      | A string value up to 10 characters                      | Returns the check number   |  |
| <b>Cnt</b>              | A string value representing a signed 32-bit integer     | Returns the transaction count of all transactions                                    |  |
| <b>CustomerID</b>       | A string value up to 50 characters                      | Returns the Customer ID of a customer to which the transaction belongs to            |  |
| <b>CV_Resp_CH</b>       | A string value up to 1 character                        | Returns the card verification result code from the payment processor                 |  |
| <b>CV_Resp_Txt_VC</b>   | A string value up to 25 characters                      | Returns the formatted response message when card verification is performed           |  |
| <b>Date_DT</b>          | A string value representing a date and time             | Returns the date on which the transaction is first made                              |  |
| <b>ERROR</b>            | A string value up to 200 characters                     | Returns an error message when a problem occurs during the transaction processing     |  |
| <b>Exp_CH</b>           | A string value up to 10 characters                      | Returns the credit card expiration date  |  |
| <b>ForceCapture</b>     | A string value representing a signed 64-bit real number | Returns the dollar amount of all ForceCapture (ForceAuth) transactions               |  |
| <b>ForceCapture_Cnt</b> | A string value representing a signed 32-bit integer     | Returns the transaction count of all ForceCapture (ForceAuth) transactions           |  |
| <b>Host_Date_CH</b>     | A string value up to 10 characters                      | Returns the payment processor's date on which the transaction is performed           |  |
| <b>Host_Ref_Num_CH</b>  | A string value up to 30 characters                      | Returns a number which uniquely identifies the transaction for the payment processor |  |
| <b>Host_Time_CH</b>     | A string value up to 10 characters                      | Returns the payment processor's time at which the transaction was performed          |  |

|                        |   |   |   |
|------------------------|---|---|---|
| <b>Invoice_ID</b>      | A string value up to 100 characters                     | Returns the transaction's Invoice number  |   |
| <b>IP_VC</b>           | A string value up to 15 characters                      | Returns the IP address of the client machine from which the transaction was processed |   |
| <b>Last_Update_DT</b>  | A string value representing a date and time             | Returns the date and time on which the transaction is last modified                   |   |
| <b>Manual</b>          | A string value representing a Boolean value             | Returns the card was swiped or not  |   |
| <b>Merchant_Key</b>    | A string value representing a signed 32-bit integer     | Returns a number which uniquely identifies a merchant                                 |   |
| <b>Name_on_Card_VC</b> | A string value up to 25 characters                      | Returns the name of the cardholder  |   |
| <b>NameOnCheck_VC</b>  | A string value up to 25 characters                      | Returns the check payer's name on the check   |   |
| <b>Orig_TRX_HD_Key</b> | A string value representing a signed 32-bit integer     | Returns the TRX_HD_Key on which the current transaction is based                      |   |
| <b>Payment_Type_ID</b> | A string value up to 10 characters                      | Returns the payment type, e.g. ECHECK   |   |
| <b>PostAuth</b>        | A string value representing a signed 64-bit real number | Returns the dollar amount of all PostAuth transactions                                |   |
| <b>PostAuth_Cnt</b>    | A string value representing a signed 32-bit integer     | Returns the transaction count of all PostAuth transactions                            |   |
| <b>Processor_ID</b>    | A string value up to 10 characters                      | Returns the name the payment processor, e.g. Vital                                    |   |
| <b>Receipt</b>         | A string value representing a signed 64-bit real number | Returns the dollar amount of all transactions with a Receipt                          | This value will always return "0"   |
| <b>Receipt_Cnt</b>     | A string value representing a signed 32-bit integer     | Returns the transaction count of all transactions with a Receipt                      | This value will always return "0"   |
| <b>Ref_Number_CH</b>   | A string  | Not currently used  | This field is not the unique transaction identifier (also called PNRRef) of the Payment Server. See the field |

|                           |   |   |                                 |
|---------------------------|---|---|---------------------------------|
|                           |   |   | TRX_HD_Key for the PNRRef value |
| <b>Register_Number_CH</b> | A string value up to 10 characters                      | Returns the register number of a transaction  |                                 |
| <b>RepeatSale</b>         | A string value representing a signed 64-bit real number | Returns the dollar amount of all RepeatSale (Recurring Billing/Installment) transactions            |                                 |
| <b>RepeatSale_Cnt</b>     | A string value representing a signed 32-bit integer     | Returns the transaction count of all RepeatSale (Recurring Billing/Installment) transactions        |                                 |
| <b>Reseller_Key</b>       | A string value representing a signed 32-bit integer     | Returns the primary key of the Reseller_T table in the database                                     |                                 |
| <b>Result_CH</b>          | A string value up to 50 characters                      | Returns the transaction processing result, e.g. 0, 12. "0" for approval, "12" for decline           |                                 |
| <b>Result_Msg_VC</b>      | A string value up to 50 characters                      | Returns the check transaction's processing result   |                                 |
| <b>Result_Msg1_VC</b>     | A string value up to 50 characters                      | Returns an extra formatted response message giving more information about the processed transaction |                                 |
| <b>Result_Msg2_VC</b>     | A string value up to 50 characters                      | Returns an extra formatted response message giving more information about the processed transaction |                                 |
| <b>Result_Txt_VC</b>      | A string value up to 150 characters                     | Returns the text message of either approval or decline for the transaction processing result        |                                 |
| <b>Return</b>             | A string value representing a signed 64-bit real number | Returns the dollar amount of all Return (Credit) transactions                                       |                                 |
| <b>Return_Cnt</b>         | A string value representing a signed 32-bit integer     | Returns the transaction count of all Return (Credit) transactions                                   |                                 |
| <b>Sale</b>               | A string value representing a signed 64-bit real number | Returns the dollar amount of all Sale transactions  |                                 |
| <b>Sale_Cnt</b>           | A string value representing a signed 32-bit integer     | Returns the transaction count of all Sale transactions  |                                 |

|                           |   |   |  |
|---------------------------|---|---|--|
| <b>Settle_Date_DT</b>     | A string value representing a date and time             | Returns the date on which the transaction is settled              |  |
| <b>Settle_Flag_CH</b>     | A string value representing a Boolean value             | Returns if the transaction is settled or not                      |  |
| <b>StateCode_CH</b>       | A string value up to 10 characters                      | Returns the state code  |  |
| <b>Street_CH</b>          | A string value up to 25 characters                      | Returns the billing street address of the credit card             |  |
| <b>SureCharge_Amt_MN</b>  | A string value representing a signed 64-bit real number | Returns the sure charge amount of a transaction                   |  |
| <b>Tip_Amt_MN</b>         | A string value representing a signed 64-bit real number | Returns the tip amount of a transaction                           |  |
| <b>Trans_Type_ID</b>      | A string value up to 20 characters                      | Returns the transaction type, e.g. Sale, Credit                   |  |
| <b>Transport_Method</b>   | A string  | Returns the Transportation Method                                 | Only for use with Dial-up transactions   |
| <b>Transport_EndPoint</b> | A string  | Returns the Transportation's Ending Destination                   | Only for use with Dial-up transactions   |
| <b>TransitNum_VC</b>      | A string value up to 200 characters                     | Returns the transit/routing number                                | This field will be masked out with asterisk (*) characters except for the last 4 digits if the System Security Level of the user is set to 1   |
| <b>TRX_Card_Key</b>       | A string value representing a signed 32-bit integer     | Returns the primary key of the TRX_Card_T table in the database   |  |
| <b>TRX_Check_Key</b>      | A string value representing a signed 32-bit integer     | Returns the primary key of the TRX_Check_T table in the database  |  |
| <b>TRX_HD_Key</b>         | A string value representing a signed 32-bit integer     | Returns the primary key of the TRX_Header_T table in the database | This field is the unique transaction identifier (also called PNRef) of the Payment Server. Use its value when submitting transactions based on a previous transaction (i.e. Voids) through the Transact.asmx Web Service |
| <b>TRX_Settle_Key</b>     | A string value representing a signed 32-bit integer     | Returns the primary key of the TRX_Settle_T table in the database |  |

|                          |   |   |  |
|--------------------------|---|---|--|
| <b>TRX_Settle_Msg_VC</b> | A string value up to 25 characters                      | Returns the transaction's settlement message  |  |
| <b>Type_CH</b>           | A string value up to 10 characters                      | Returns the credit card type, e.g. VISA, MASTERCARD                                     |  |
| <b>User_Name_VC</b>      | A string value up to 25 characters                      | Returns the username, under which the transactions were made                            |  |
| <b>Void_Flag_CH</b>      | A string value representing a Boolean value             | Returns the transaction is voided or not  |  |
| <b>Zip_CH</b>            | A string value up to 10 characters                      | Returns the billing zip code of the credit card   |  |
| <b>Auth_Amt_MN</b>       | A string value representing a signed 64-bit real number | Returns the authorized amount of a card transaction                                     |  |
| <b>Authorization</b>     | A string value representing a signed 64-bit real number | Returns the dollar amount of all Authorization (PreAuth) transactions                   |  |
| <b>Authorization_Cnt</b> | A string value representing a signed 32-bit integer     | Returns the transaction count of all Authorization (PreAuth) transactions               |  |
| <b>AVS_Resp_CH</b>       | A string value up to 1 character                        | Returns the address verification result code from the payment processor                 |  |
| <b>AVS_Resp_Txt_VC</b>   | A string value up to 25 characters                      | Returns the formatted response message when address verification is performed           |  |
| <b>Batch_Number</b>      | A string value up to 10 characters                      | Returns the batch number for the transaction that was returned by the payment processor | Not all payment processors support returning this data element |
| <b>Capture</b>           | A string value representing a signed 64-bit real number | Returns the dollar amount of all Capture transactions                                   | This value will always return "0"                              |
| <b>Capture_Cnt</b>       | A string value representing a signed 32-bit integer     | Returns the transaction count of all Capture transactions                               | This value will always return "0"                              |
| <b>Card_Info_Key</b>     | A string value representing a signed 32-bit integer     | Returns the primary key of the CC_Info_T table in the database                          |  |
| <b>Cash_Back_Amt_MN</b>  | A string value representing a signed 64-bit real number | Returns the cash back amount for a debit or EBT transaction                             |  |

|                         |   |  |  |
|-------------------------|---|--|--|
| <b>CheckNum_CH</b>      | A string value up to 10 characters                      | Returns the check number   |  |
| <b>Cnt</b>              | A string value representing a signed 32-bit integer     | Returns the transaction count of all transactions                                    |  |
| <b>CustomerID</b>       | A string value up to 50 characters                      | Returns the Customer ID of a customer to which the transaction belongs to            |  |
| <b>CV_Resp_CH</b>       | A string value up to 1 character                        | Returns the card verification result code from the payment processor                 |  |
| <b>CV_Resp_Txt_VC</b>   | A string value up to 25 characters                      | Returns the formatted response message when card verification is performed           |  |
| <b>Date_DT</b>          | A string value representing a date and time             | Returns the date on which the transaction is first made                              |  |
| <b>ERROR</b>            | A string value up to 200 characters                     | Returns an error message when a problem occurs during the transaction processing     |  |
| <b>Exp_CH</b>           | A string value up to 10 characters                      | Returns the credit card expiration date  |  |
| <b>ForceCapture</b>     | A string value representing a signed 64-bit real number | Returns the dollar amount of all ForceCapture (ForceAuth) transactions               |  |
| <b>ForceCapture_Cnt</b> | A string value representing a signed 32-bit integer     | Returns the transaction count of all ForceCapture (ForceAuth) transactions           |  |
| <b>Host_Date_CH</b>     | A string value up to 10 characters                      | Returns the payment processor's date on which the transaction is performed           |  |
| <b>Host_Ref_Num_CH</b>  | A string value up to 30 characters                      | Returns a number which uniquely identifies the transaction for the payment processor |  |
| <b>Host_Time_CH</b>     | A string value up to 10 characters                      | Returns the payment processor's time at which the transaction was performed          |  |
| <b>Invoice_ID</b>       | A string value up to 100                                | Returns the  |  |

|                        |   |   |  |
|------------------------|---|---|--|
|                        | characters  | transaction's Invoice number  |  |
| <b>IP_VC</b>           | A string value up to 15 characters                      | Returns the IP address of the client machine from which the transaction was processed |  |
| <b>Last_Update_DT</b>  | A string value representing a date and time             | Returns the date and time on which the transaction is last modified                   |  |
| <b>Manual</b>          | A string value representing a Boolean value             | Returns the card was swiped or not  |  |
| <b>Merchant_Key</b>    | A string value representing a signed 32-bit integer     | Returns a number which uniquely identifies a merchant                                 |  |
| <b>Name_on_Card_VC</b> | A string value up to 25 characters                      | Returns the name of the cardholder  |  |
| <b>NameOnCheck_VC</b>  | A string value up to 25 characters                      | Returns the check payer's name on the check   |  |
| <b>Orig_TRX_HD_Key</b> | A string value representing a signed 32-bit integer     | Returns the TRX_HD_Key on which the current transaction is based                      |  |
| <b>Payment_Type_ID</b> | A string value up to 10 characters                      | Returns the payment type, e.g. ECHECK   |  |
| <b>PostAuth</b>        | A string value representing a signed 64-bit real number | Returns the dollar amount of all PostAuth transactions                                |  |
| <b>PostAuth_Cnt</b>    | A string value representing a signed 32-bit integer     | Returns the transaction count of all PostAuth transactions                            |  |
| <b>Processor_ID</b>    | A string value up to 10 characters                      | Returns the name the payment processor, e.g. Vital                                    |  |
| <b>Receipt</b>         | A string value representing a signed 64-bit real number | Returns the dollar amount of all transactions with a Receipt                          | This value will always return "0"  |
| <b>Receipt_Cnt</b>     | A string value representing a signed 32-bit integer     | Returns the transaction count of all transactions with a Receipt                      | This value will always return "0"  |
| <b>Ref_Number_CH</b>   | A string  | Not currently used  | This field is not the unique transaction identifier (also called PNRRef) of the Payment Server. See the field TRX_HD_Key for the |

|                           |   |   |             |
|---------------------------|---|---|-------------|
|                           |   |   | PNRef value |
| <b>Register_Number_CH</b> | A string value up to 10 characters                      | Returns the register number of a transaction  |             |
| <b>RepeatSale</b>         | A string value representing a signed 64-bit real number | Returns the dollar amount of all RepeatSale (Recurring Billing/Installment) transactions            |             |
| <b>RepeatSale_Cnt</b>     | A string value representing a signed 32-bit integer     | Returns the transaction count of all RepeatSale (Recurring Billing/Installment) transactions        |             |
| <b>Reseller_Key</b>       | A string value representing a signed 32-bit integer     | Returns the primary key of the Reseller_T table in the database                                     |             |
| <b>Result_CH</b>          | A string value up to 50 characters                      | Returns the transaction processing result, e.g. 0, 12. "0" for approval, "12" for decline           |             |
| <b>Result_Msg_VC</b>      | A string value up to 50 characters                      | Returns the check transaction's processing result   |             |
| <b>Result_Msg1_VC</b>     | A string value up to 50 characters                      | Returns an extra formatted response message giving more information about the processed transaction |             |
| <b>Result_Msg2_VC</b>     | A string value up to 50 characters                      | Returns an extra formatted response message giving more information about the processed transaction |             |
| <b>Result_Txt_VC</b>      | A string value up to 150 characters                     | Returns the text message of either approval or decline for the transaction processing result        |             |
| <b>Return</b>             | A string value representing a signed 64-bit real number | Returns the dollar amount of all Return (Credit) transactions                                       |             |
| <b>Return_Cnt</b>         | A string value representing a signed 32-bit integer     | Returns the transaction count of all Return (Credit) transactions                                   |             |
| <b>Sale</b>               | A string value representing a signed 64-bit real number | Returns the dollar amount of all Sale transactions  |             |
| <b>Sale_Cnt</b>           | A string value representing a signed 32-bit integer     | Returns the transaction count of all Sale transactions  |             |

|                           |   |   |  |
|---------------------------|---|---|--|
| <b>Settle_Date_DT</b>     | A string value representing a date and time             | Returns the date on which the transaction is settled              |  |
| <b>Settle_Flag_CH</b>     | A string value representing a Boolean value             | Returns if the transaction is settled or not                      |  |
| <b>StateCode_CH</b>       | A string value up to 10 characters                      | Returns the state code  |  |
| <b>Street_CH</b>          | A string value up to 25 characters                      | Returns the billing street address of the credit card             |  |
| <b>SureCharge_Amt_MN</b>  | A string value representing a signed 64-bit real number | Returns the sure charge amount of a transaction                   |  |
| <b>Tip_Amt_MN</b>         | A string value representing a signed 64-bit real number | Returns the tip amount of a transaction                           |  |
| <b>Total_Amt_MN</b>       | A string value representing a signed 64-bit real number | Returns the total amount of a transaction                         |  |
| <b>Trans_Type_ID</b>      | A string value up to 20 characters                      | Returns the transaction type, e.g. Sale, Credit                   |  |
| <b>Transport_Method</b>   | A string  | Returns the Transportation Method                                 | Only for use with Dial-up transactions   |
| <b>Transport_EndPoint</b> | A string  | Returns the Transportation's Ending Destination                   | Only for use with Dial-up transactions   |
| <b>TransitNum_VC</b>      | A string value up to 200 characters                     | Returns the transit/routing number                                | This field will be masked out with asterisk (*) characters except for the last 4 digits if the System Security Level of the user is set to 1   |
| <b>TRX_Card_Key</b>       | A string value representing a signed 32-bit integer     | Returns the primary key of the TRX_Card_T table in the database   |  |
| <b>TRX_Check_Key</b>      | A string value representing a signed 32-bit integer     | Returns the primary key of the TRX_Check_T table in the database  |  |
| <b>TRX_HD_Key</b>         | A string value representing a signed 32-bit integer     | Returns the primary key of the TRX_Header_T table in the database | This field is the unique transaction identifier (also called PNRef) of the Payment Server. Use its value when submitting transactions based on a previous transaction (i.e. Voids) through the Transact.asmx Web Service |

|                          |   |   |  |
|--------------------------|---|---|--|
| <b>TRX_Settle_Key</b>    | A string value representing a signed 32-bit integer | Returns the primary key of the TRX_Settle_T table in the database |  |
| <b>TRX_Settle_Msg_VC</b> | A string value up to 25 characters                  | Returns the transaction's settlement message                      |  |
| <b>Type_CH</b>           | A string value up to 10 characters                  | Returns the credit card type, e.g. VISA, MASTERCARD               |  |
| <b>User_Name_VC</b>      | A string value up to 25 characters                  | Returns the username, under which the transactions were made      |  |
| <b>Void_Flag_CH</b>      | A string value representing a Boolean value         | Returns the transaction is voided or not                          |  |
| <b>Zip_CH</b>            | A string value up to 10 characters                  | Returns the billing zip code of the credit card                   |  |

## Response Values

### ***Result Response Field Definitions (Error Codes)***

The list below contains result codes returned in the Result response field of the XMLPayResponse when using a transaction processing Transact.asmx web service operation (i.e. ProcessCreditCard, ProcessCheck, etc). A decline returned by the payment processor for this response field is value twelve (12) or thirteen (13). An approval is value zero (0). Any other value is an error code, which is returned by the payment gateway and not by the payment processor. Please note that when programmatically validating a transaction's result, this value should be used instead of any response message describing the result. I.e. do *not* use RespMSG or Message response fields, as these values may vary. Please note that this list is subject to change without prior notice.

| <b>Value</b> | <b>Description</b>                            |
|--------------|---|
| -100         | Transaction NOT Processed; Generic Host Error |
| 0            | Approved                                      |
| 1            | User Authentication Failed                    |
| 2            | Invalid Transaction                           |
| 3            | Invalid Transaction Type                      |
| 4            | Invalid Amount                                |

|     |  |
|-----|--|
| 5   | Invalid Merchant Information   |
| 7   | Field Format Error   |
| 8   | Not a Transaction Server   |
| 9   | Invalid Parameter Stream   |
| 10  | Too Many Line Items  |
| 11  | Client Timeout Waiting for Response  |
| 12  | Decline  |
| 13  | Referral   |
| 14  | Transaction Type Not Supported In This Version   |
| 19  | Original Transaction ID Not Found  |
| 20  | Customer Reference Number Not Found  |
| 22  | Invalid ABA Number   |
| 23  | Invalid Account Number   |
| 24  | Invalid Expiration Date  |
| 25  | Transaction Type Not Supported by Host   |
| 26  | Invalid Reference Number   |
| 27  | Invalid Receipt Information  |
| 28  | Invalid Check Holder Name  |
| 29  | Invalid Check Number   |
| 30  | Check DL Verification Requires DL State  |
| 40  | Transaction did not connect (to NCN because SecureNCIS is not running on the web server) |
| 50  | Insufficient Funds Available   |
| 99  | General Error  |
| 100 | Invalid Transaction Returned from Host   |
| 101 | Timeout Value too Small or Invalid Time Out Value  |
| 102 | Processor Not Available  |
| 103 | Error Reading Response from Host   |
| 104 | Timeout waiting for Processor Response   |
| 105 | Credit Error   |
| 106 | Host Not Available   |
| 107 | Duplicate Suppression Timeout  |

|      |  |
|------|--|
| 108  | Void Error                                   |
| 109  | Timeout Waiting for Host Response            |
| 110  | Duplicate Transaction                        |
| 111  | Capture Error                                |
| 112  | Failed AVS Check                             |
| 113  | Cannot Exceed Sales Cap                      |
| 1000 | Generic Host Error                           |
| 1001 | Invalid Login                                |
| 1002 | Insufficient Privilege or Invalid Amount     |
| 1003 | Invalid Login Blocked                        |
| 1004 | Invalid Login Deactivated                    |
| 1005 | Transaction Type Not Allowed                 |
| 1006 | Unsupported Processor                        |
| 1007 | Invalid Request Message                      |
| 1008 | Invalid Version                              |
| 1010 | Payment Type Not Supported                   |
| 1011 | Error Starting Transaction                   |
| 1012 | Error Finishing Transaction                  |
| 1013 | Error Checking Duplicate                     |
| 1014 | No Records To Settle (in the current batch)  |
| 1015 | No Records To Process (in the current batch) |

## ***AVS Response Codes***

The following table contains the possible response values returned for address verification (AVS).

| <b>Value</b> | <b>Description</b>                      |
|--------------|---|
| X            | Exact: Address and nine-digit Zip match |
| Y            | Yes: Address and five-digit Zip match   |

|   |   |
|---|---|
| A | Address: Address matches, Zip does not  |
| Z | 5-digit Zip: 5-digit Zip matches, address doesn't   |
| W | Whole Zip: 9-digit Zip matches, address doesn't   |
| N | No: Neither address nor Zip matches   |
| U | Unavailable: Address information not available  |
| G | Unavailable: Address information not available for international transaction  |
| R | Retry: System unavailable or time-out   |
| E | Error: Transaction unintelligible for AVS or edit error found in the message that prevents AVS from being performed |
| S | Not Supported: Issuer doesn't support AVS service   |
| B | * Street Match: Street addresses match for international transaction, but postal code doesn't                       |
| C | * Street Address: Street addresses and postal code not verified for international transaction                       |
| D | * Match: Street addresses and postal codes match for international transaction                                      |
| I | * Not Verified: Address Information not verified for International transaction                                      |
| M | * Match: Street addresses and postal codes match for international transaction                                      |
| P | * Postal Match: Postal codes match for international transaction, but street address doesn't                        |
| 0 | ** No response sent   |
| 5 | Invalid AVS response  |

\* These values are Visa specific.

\*\* These values are returned by the Payment Server and not the processor.

## ***CV Response Codes***

The following table contains the possible response values returned for a CVV2/CVC2/CID check.

| <b>Value</b> | <b>Description</b>     |
|--------------|------------------------|
| M            | CVV2/CVC2/CID Match    |
| N            | CVV2/CVC2/CID No Match |
| P            | Not Processed          |

|   |  |
|---|--|
| S | Issuer indicates that the CV data should be present on the card, but the merchant has indicated that the CV data is not present on the card. |
| U | Unknown / Issuer has not certified for CV or issuer has not provided Visa/MasterCard with the CV encryption keys.                            |
| X | Server Provider did not respond  |

## Valid Parameter Input Characters

The table below displays all allowable characters (unless otherwise noted) that are accepted by the Payment Server. Characters are displayed in Courier New font. All other characters may cause undesirable results.

**Table 1. Valid Data Characters**

| DEC | HEX | Character | DEC | HEX | Character | DEC | HEX | Character |
|-----|-----|-----------|-----|-----|-----------|-----|-----|-----------|
| 32  | 20  | Space     | 63  | 3F  | ?         | 96  | 60  | `         |
| 33  | 21  | !         | 64  | 40  | @         | 97  | 61  | a         |
| 34  | 22  | "         | 65  | 41  | A         | 98  | 62  | b         |
| 35  | 23  | #         | 66  | 42  | B         | 99  | 63  | c         |
| 36  | 24  | \$        | 67  | 43  | C         | 100 | 64  | d         |
| 37  | 25  | %         | 68  | 44  | D         | 101 | 65  | e         |
| 38  | 26  | &         | 69  | 45  | E         | 102 | 66  | f         |
| 39  | 27  | '         | 70  | 46  | F         | 103 | 67  | g         |
| 40  | 28  | (         | 71  | 47  | G         | 104 | 68  | h         |
| 41  | 29  | )         | 72  | 48  | H         | 105 | 69  | i         |
| 42  | 2A  | *         | 73  | 49  | I         | 106 | 6A  | j         |
| 43  | 2B  | +         | 74  | 4A  | J         | 107 | 6B  | k         |
| 44  | 2C  | ,         | 75  | 4B  | K         | 108 | 6C  | l         |
| 45  | 2D  | -         | 76  | 4C  | L         | 109 | 6D  | m         |
| 46  | 2E  | .         | 77  | 4D  | M         | 110 | 6E  | n         |
| 47  | 2F  | /         | 78  | 4E  | N         | 111 | 6F  | o         |
| 48  | 30  | 0         | 79  | 4F  | O         | 112 | 70  | p         |
| 49  | 31  | 1         | 80  | 50  | P         | 113 | 71  | q         |
| 50  | 32  | 2         | 81  | 51  | Q         | 114 | 72  | r         |
| 51  | 33  | 3         | 82  | 52  | R         | 115 | 73  | s         |
| 52  | 34  | 4         | 83  | 53  | S         | 116 | 74  | t         |
| 53  | 35  | 5         | 84  | 54  | T         | 117 | 75  | u         |
| 54  | 36  | 6         | 85  | 55  | U         | 118 | 76  | v         |
| 55  | 37  | 7         | 86  | 56  | V         | 119 | 77  | w         |
| 56  | 38  | 8         | 87  | 57  | W         | 120 | 78  | x         |
| 57  | 39  | 9         | 88  | 58  | X         | 121 | 79  | y         |
| 58  | 3A  | :         | 89  | 59  | Y         | 122 | 7A  | z         |

|    |    |   |    |    |   |     |    |   |
|----|----|---|----|----|---|-----|----|---|
| 59 | 3B | ; | 90 | 5A | Z | 123 | 7B | { |
| 60 | 3C | < | 92 | 5C | \ | 124 | 7C |   |
| 61 | 3D | = | 94 | 5E | ^ | 125 | 7D | } |
| 62 | 3E | > | 95 | 5F | _ | 126 | 7E | ~ |

## Character Removal

The table below displays all acceptable characters that must be removed by the Payment Server before submitting information to the Web Service operations. See each input parameter for each Web Service operation in order to know which input parameters will have these characters removed. This character removal ensures that the Payment Servers' internal XML parsers can properly read the information of the Web Service operation. Characters in the table are displayed in Courier New font.

Many XML Parsers will encode these characters for you. In this case, the characters *will not* be converted back to their proper values by the Payment Server; they will be taken literally. Also, if you pass the encoded character through an input parameter that removes the characters listed in the table below, then certain characters may be removed (see examples below). However, if you are not using a parser, or if the parser does not handle this encoding, then the characters in the table listed below may still be removed, depending on the input parameter for the Web Service operation you are using.

**Table 2. XML Character Removal**

| Character | XML Parser Encoding |
|-----------|---------------------|
| <         | &lt;                |
| >         | &gt;                |
| &         | &amp;               |
| '         | &apos;              |
| "         | &quot;              |

### Example

The following example shows how characters would be removed if the data was passed through the NameOnCard parameter of the ProcessCreditCard operation.

Valid: *John James*

Invalid: *John & James becomes John James*

Invalid: *John &amp; James becomes John amp; James*