



Web Transactions Services

AHEAD OF THE COMPETITION, BEHIND YOUR BUSINESS!

Merchant Integration Guide

Version v1.3

Revision History

| | |
|----------|---|
| 02/08/05 | Created Documentation (Erik Redding) |
| 03/21/05 | General edits: Added field lengths for DBS section, added field lengths for signup join form. (Erik Redding) |
| 3/23/05 | General edits: Membership upgrade for Signup separated (Erik Redding) |
| 06/17/05 | General edits: PostBack Moved down and Join page moved up. |
| 8/29/05 | Added cross sale postback information. |

Table Of Contents

| | |
|--|-----------|
| Revision History | 2 |
| Signup | 4 |
| 1. Overview | 4 |
| 2. Join Form | 4 |
| 2.1 Form Pre-population | 4 |
| Table 2.1.1: Allowed fields for pre-population | 4 |
| 2.2 Viewing the Join Form | 4 |
| 2.3 Additional Join Form Information | 5 |
| 3. Postback | 5 |
| 3.1 Postback Description and Response Codes..... | 5 |
| Example 3.1.1 Response Codes Pseudo Code:..... | 5 |
| 3.2 Initial Postback | 5 |
| Table 3.2.1: Initial Post-back Variables | 5 |
| 3.3 Second (Sale) Postback | 6 |
| Table 3.3.1 Second (Sale) Postback Variables..... | 6 |
| 3.4 Delete Postback | 7 |
| Table 3.4.1: Delete postback variables | 7 |
| 3.5 Membership Upgrade Postback: | 7 |
| Table 3.5.1: Membership upgrade variables | 7 |
| 3.6 Cross Sale Postback: | 7 |
| Table 3.6.1 Cross Sale Postback Variables | 7 |
| 3.7 Additional Postbacks | 8 |
| Order Refresh Requests | 8 |
| 4. Membership Upgrade..... | 9 |
| 4.1 Overview..... | 9 |
| 4.2 Membership Upgrade URL Fields: | 9 |
| Table 4.2.1: Membership upgrade variables | 9 |
| 4.3 Using Membership Upgrade | 9 |
| Transaction History Files | 10 |
| 1. Overview: | 10 |
| 2. Transaction History File Format | 10 |
| Table 2.1.1: Transaction History File Fields | 10 |
| 3. Example Data..... | 11 |
| 4. Order Tracking | 12 |
| ACH Transaction Types | 13 |
| ACH Transaction Life Cycle | 13 |
| Additional Resources | 14 |

Signup

1. Overview

The Signup system provides merchants with ACH processing with minimal programming and a high degree of customization. Join form pre-population, transaction related postbacks, and transaction history files highlight the main features that allow merchants to effectively track subscription-based consumers. Our recent addition of the Membership Upgrade capabilities allows merchants to offer initial trial member consumers an opportunity to upgrade early, therefore raising retention and conversions to recurring memberships. WTS provides merchants with an account ParentID and SubID's to define specific products.

The signup process follows this order:

1. Consumer initiated join form request from Merchant Site
2. Consumer will fill out the form and click submit:
 - a. Post-back to merchant (action = exist)
 - a1. If username exists; return to join form;
 - a2. Otherwise, continue
 - b. If the account is valid, Post-back to Merchant to add user, activating account(action = add);
If the account is invalid, no second post-back is sent;
3. User receives result of transaction, either accepted or declined.

2. Join Form

2.1 Form Pre-population

The following form fields can be pre-populated through a GET string:

Table 2.1.1: Allowed fields for pre-population

| Form Field Name: | Example: | Max Field Length: | Description |
|------------------|----------------------|-------------------|----------------------------|
| custname | Your Name | 30 | Consumer name |
| custemail | Your Email Address | 50 | Consumer address |
| custstate | State | 2 | Consumer state |
| Zip | ZIP | 6 | Consumer zip |
| zip2 | | 5 | Consumer zip extra |
| username | Choose Username | 50 | Consumer username |
| password | Select your Password | 50 | Consumer password |
| passwordv | Verify your Password | 50 | Password verification |
| Profileid | Membership Plan | 6 | Pre-selects billing option |

2.2 Viewing the Join Form

The join form can be viewed by the following format:

<https://join.achbill.com/cgi-bin/signup.cgi?chk:wts01:<SubID>:extra:>

An example:

https://join.achbill.com/cgi-bin/signup.cgi?chk:wts01:MERCTEST:webmaster_extra_code:&custname=John+Doe&custemail=john@johndoe.com&custstate=TX&zip=12345&zip2=1234&username=jdoesexe&password=jerry&passwordv=jerry&profileid=123456

2.3 Additional Join Form Information

- The join form can be called through a GET request by using the variable **cfginfo** with the first portion of the GET string above that is separated by colons. The format follows this example:

```
<name="cfginfo" type="hidden" value="chk:WTS01:<subID>:extrainfo">
```

- We can change the default min and max lengths for the username and password fields. By default these are set to 4 (min) to 20 (max). The max length is 32.

3. Postback

3.1 Postback Description and Response Codes

The postback is used primarily for user account maintenance. The postback does contain transaction specific data, but only on the second post-back. A postback will send a POST variable **action** to indicate the process that is occurring. **action** will contain one of the following requests:

```
action = exist
action = add
action = delete
```

Custom post-back scripts should respond as outlined in the pseudo code example 2.1.1:

Example 3.1.1 Response Codes Pseudo Code:

```
if action == exists
  print *failed* -- username does not exist
else
  print *success* -- username does exist

if action == add
  print *success* -- username added
else
  print *error* -- username not added $some_error_message

if action == delete
  print *success* -- username deleted
else
  print *error* -- username not deleted
```

3.2 Initial Postback

The first postback is sent to check if a username exists. If it exists the transaction is stopped, the user is returned to the join form and asked to specify a different username then the process is restarted. The following variables are sent during the initial postback:

Table 3.2.1: Initial Post-back Variables

| Variable: | Example: | Description: |
|-----------|------------------|---|
| username | Jerry | Desired username |
| Siteid | MERCTEST | WTS assigned SubID for the site |
| sys_pass | systempassword32 | WTS assigned system password |
| Site | MERCTEST | Same as siteid |
| Action | exists | Action requests check to see if user exists |

3.3 Second (Sale) Postback

The second postback will be sent to add a username if the account has been pre-authorized for the initial amount of the transaction. The following table outlines the details of the second postback:

Table 3.3.1 Second (Sale) Postback Variables

| Field: | Data: | Description: |
|------------------|------------------------|--|
| result | 1 | 1 = The transaction result was a success anything else means that there was a problem |
| orderid | 7654321 | Unique key that is unique for every account |
| authno | CHK Pre-Auth:007654321 | Information received from Customer's bank stating that the amount is Pre-Authorized |
| custname | John Doe | First and Last Name of customer |
| address1 | 123 Mary Poppins Dr | The Street Address of the customer |
| address2 | | Continuation of the address |
| city | someCity | Customer's home city |
| state | TX | Customer's home State |
| country | | Country in which the customer lives if outside of U.S. |
| zip1 | 12345 | The Zip Code |
| password | password | Customer's password |
| email | john@someemail.com | Customer's Email Address |
| required_clicked | CHECKED | CHECKED means that the users accepted the Terms and Conditions, The transaction will be stopped if the user did not accept |
| siteid | TESTIT | SubID assigned by WTS |
| site | TESTIT | SubID assigned by WTS (contains same data as siteid) |
| orderinfo | | Data that was appended to the end of the signup url (usually affiliate codes are appended there) |
| extra | | Data that was appended to the end of the url. It will contain the same data as orderinfo DEPRECATED |
| pmt_type | chk | This will always be chk for checking account |
| amount | 495 | Amount of transaction in cents does not include decimals or commas |
| currency | US | Type of currency for billing will be either US for U.S. Currency or CN for Canadian Currency |
| sitename | yourDomain | Domain Name of your site |
| memtype | 00001 | Unique key that identifies the Billing Profile used by the customer |
| historyid | 7654321 | Unique key that is assigned to every transaction. |
| referrer | | The Server Referral information that was provided |
| action | add | Tells the script to add this username. |
| sys_pass | systemPassword | System Password used by WTS for User Account Maintenance |
| ipaddress | 12.123.12.298 | Customer's IP Address |
| reseller_code | 123456 | Sent on postback for cross sales only contains the reseller_code of the company sending the cross sale |
| email | john@someemail.com | Customer's Email Address |

3.4 Delete Postback

The action=delete request will contain each field the initial action=exist postback does, but will contain action=delete instead.

Table 3.4.1: Delete postback variables

| Variable: | Example: | Description |
|-----------|------------------|--|
| username | Jerry | Desired username |
| siteid | MERCTEST | WTS assigned SubID for the site |
| sys_pass | systempassword32 | WTS assigned system password for user account maintenance. |
| site | MERCTEST | Same as siteid |
| action | delete | Delete the requested username. |

3.5 Membership Upgrade Postback:

The postback for a membership upgrade will include all of the Second (Sale) Postback variables and the variables listed in Table 4.2.1.

Table 3.5.1: Membership upgrade variables

| Variable: | Example: | Description: |
|---------------|--------------------|---|
| pmt_type | upg | Payment Type of Upgrade |
| memtype | 21768 | This will be the Billing ProfileID/ProfileKeyID that we assigned to you. Each billing option has its own unique memtype. A unique field given to each transaction. This is given to you in the postback and in the transaction history files we send out nightly. |
| consumer_code | QQmA638AAAEACiSNCl | This is the order_id of the order that is being upgraded. This is the primary requirement for any upgrade. |
| order_id | 1234567 | |

3.6 Cross Sale Postback:

The postback for a Cross Sale will include the following variables.

Table 3.6.1 Cross Sale Postback Variables

| Field: | Data: | Description: |
|------------------|------------------------|--|
| result | 1 | 1 = The transaction result was a success anything else means that there was a problem |
| orderid | 7654321 | Unique key that is unique for every account |
| authno | CHK Pre-Auth:007654321 | Information received from Customer's bank stating that the amount is Pre-Authorized |
| custname | John Doe | First and Last Name of customer |
| address1 | 123 Mary Poppins Dr | The Street Address of the customer |
| address2 | | Continuation of the address |
| city | someCity | Customer's home city |
| state | TX | Customer's home State |
| password | password | Customer's password |
| email | john@someemail.com | Customer's Email Address |
| required_clicked | CHECKED | CHECKED means that the users accepted the Terms and Conditions, The transaction will be stopped if the user did not accept |
| siteid | TESTIT | SubID assigned by WTS |
| site | TESTIT | SubID assigned by WTS (contains same data as siteid) |
| orderinfo | | Data that was appended to the end of the signup url (usually affiliate codes are appended there) |
| extra | | Data that was appended to the end of the url. It will contain the same data as orderinfo DEPRECATED |

| | | |
|---------------|--------------------|--|
| pmt_type | chk | This will always be chk for checking account |
| amount | 495 | Amount of transaction in cents does not include decimals or commas |
| memtype | 00001 | Unique key that identifies the Billing Profile used by the customer |
| historyid | 7654321 | Unique key that is assigned to every transaction. |
| referrer | | The Server Referral information that was provided |
| action | add | Tells the script to add this username. |
| sys_pass | systemPassword | System Password used by WTS for User Account Maintenance |
| reseller_code | 123456 | Sent on postback for cross sales only contains the reseller_code of the company sending the cross sale |
| email | john@someemail.com | Customer's Email Address |

3.7 Additional Postbacks

Order Refresh Requests

In the event of a postback error, the customer support department will send a refresh postback. This is indicated by cs=wts_refresh. This is only to re-post the transaction if there's been an issue during signup.

4. Membership Upgrade

4.1 Overview

Membership Upgrade is available to merchants who need a solution for paid limited access to their subscription-based sites. The membership upgrade gives the merchants the ability to upgrade the trial *before* the trial period is over, therefore keeping retention as high as possible. The table below outlines fields required for a membership upgrade.

4.2 Membership Upgrade URL Fields:

Table 4.2.1: Membership upgrade variables

| Variable: | Example: | Status: | Description: |
|---------------|----------------------|--------------|--|
| pmt_type | upg | Required | Payment Type of Upgrade |
| memtype | 21768 | Not required | This will be the Billing ProfileID/ProfileKeyID that we assigned to you. Each billing option has its own unique memtype. |
| consumer_code | QQmA638AAAEAAACiSNCI | Not required | A unique field given to each transaction. This is given to you in the postback and in the transaction history files we send out nightly. |
| order_id | 1234567 | Required | This is the order_id of the order that is being upgraded. This is the primary requirement for any upgrade. |

4.3 Using Membership Upgrade

To offer the consumer the opportunity to upgrade, you will use the following format to call the upgrade form:

`https://join.achbill.com/cgi-bin/signup.cgi?upg:wts01:<SubID>:&order_id=orderidtoupgrade`

An example:

`https://join.achbill.com/cgi-bin/signup.cgi?upg:wts01:TEST03&order_id=6389605`

Transaction History Files

1. Overview:

The transaction history files contain all initial sales (Check Pre-Note / Check Pre-Auth), Returns (Check Return), Money Received (Check Settlement), and Refunds (Check Refund) from the previous day. A transaction history file will come in a flat, quote-qualifier, comma-delimited file, which can either be picked up (via provided IP address) or sent to merchant servers.

The following Operating system are expecting the following to know when there is an end of line:

UNIX uses a (LF) LineFeed

Windows uses a (CRLF) Carriage Return / Line Feed

The Transaction History File on our server will only have a (LF) Line Feed

If you transfer the file correctly using ASCII on a Windows Machine you will get a file with (CRLF) Carriage Return / Line Feed

If transferred correct (ASCII) it will have CRLF on windows

If Transferred incorrectly the file in BINARY the file will looked garbled and all data will be on the first line.

It is imperative that you download the file in ASCII to ensure there is no data corruption.

2. Transaction History File Format

The naming format of the transaction history file will be:

PARENTID-trans-WTS-YYYYMMDD.txt e.g.: **WTSTEST-trans-WTS-20050212.txt**

The files will contain the following fields listed in Table 2.1:

Table 2.1.1: Transaction History File Fields

| Field: | Description: |
|--------------------------|--|
| SubID | The SubID assigned by WTS |
| Transaction Date | Date of the transaction |
| Amount | Amount of the transaction |
| Consumer Name | Customer's full name |
| Account Name | Customer's account name |
| Transaction Type | Details the transaction type Check Pre-Auth for preauthorization, Check Return for returned check, Check Settlement for funds that have been received. |
| Transaction Result | Details whether the transaction was approved or declined |
| Authorization Code | This is the code that we received from the customer's bank |
| Account Type Description | Will always be check |
| Recurring Description | will be initial or recurring |
| Company Name | Company Name if given during transaction |
| Billing Address | Customer Mailing information |
| Billing Address2 | |
| Billing City | |
| Billing State | |
| Billing Zip | |
| Billing Country | |
| Shipping Address | |
| Shipping Address2 | |

| | |
|---------------------|--|
| Shipping City | |
| Shipping State | |
| Shipping Zip | |
| Shipping Country | |
| Phone Number | Customer Phone Number |
| E-Mail Address | Customer E-mail Address |
| IP Address | The IP address of the customer |
| Server Referrer | This will contain referrer information that was submitted during transaction |
| MerchantOrderNumber | contains any extra affiliate code information submitted during transaction |
| Order Number | Unique key assigned to every order |
| History KeyID | Unique key associated with each transaction of an order |
| Reference KeyID | Contains the previous History Keyid |
| Profile KeyID | If a billing profile keyid was provided it will be listed here |
| Reseller Code | Used for cross sell transactions |
| Partner Code | Will contain the partner associated with this transaction |
| Username | If username is sent we will include it here |
| ConsumerUniqueID | Will be used later for offering One-Click sales to current/former customers |

The file format should be in the order listed above, but here is each field inside example delimiting fields:

"SubID","Transaction Date","Amount","Consumer Name","Account Name","Transaction Type","Transaction Result","Authorization Code","Routing Number","Account Number","Account Type Description","Credit Card Number","Credit Card Expiration Date","Recurring Description","Company Name","Billing Address","Billing Address2","Billing City","Billing State","Billing Zip","Billing Country","Shipping Address","Shipping Address2","Shipping City","Shipping State","Shipping Zip","Shipping Country","Phone Number","E-Mail Address","IP Address","Server Referrer","MerchantOrderNumber","Order Number","History KeyID","Reference KeyID","Profile KeyID","Reseller Code","Partner Code","Username","ConsumerUniqueID"

3. Example Data

Note: There will be no word rap in the Transaction history files, therefore each example listed below will actually be on one line.

Check Pre-Note:

"WTS01","Jun 28, 2003 12:03AM","6.95","John Doe","John Doe","Check Pre-Note","Approved","CheckAuth:009999999","HIDDEN","HIDDEN","Check","","","Initial","","123 JohnDoe st","","Johnson City","TX","12345","","","","","(123)123-4567","johndoe@website.com","123.123.123.123","","1000","1234567","1234567","","12345","",""

Check Pre-Auth:

"WTS01","Jun 28, 2003 12:03AM","6.95","John Doe","John Doe","Check Pre-Auth","Approved","CheckAuth:009999999","HIDDEN","HIDDEN","Check","","","Initial","","123 JohnDoe st","","Johnson City","TX","12345","","","","","(123)123-4567","johndoe@website.com","123.123.123.123","","1000","1234567","1234567","","12345","",""

Check Settlement:

"WTS02","Jun 28, 2003 02:21AM","6.95","John Doe","John Doe","Check Settlement","Approved","CheckAuth:009999999","HIDDEN","HIDDEN","Check","","","Initial","","123 JohnDoe st","","Johnson City","TX","12345","","","","","(123)123-4567","johndoe@website.com","123.123.123.123","","1000","1234567","1234567","","12345","",""

Check Late Return:

"WTS02","Jun 28, 2003 02:21AM","6.95","John Doe","John Doe","Check Late Return","Declined","Customer Advises: Not Authorized","HIDDEN","HIDDEN","Check","","","Initial","","123 JohnDoe st","","Johnson City","TX","12345","","","","","(123)123-4567","johndoe@website.com","123.123.123.123","","1000","1234567","1234567","","12345","",""

Check Return:

"WTS03","Jun 28, 2003 02:27AM","6.95","John Doe","John Doe","Check Return","Declined","NSF","HIDDEN","HIDDEN","Check","","","Initial","","123 JohnDoe st","","Johnson City","TX","12345","","","","","(123)123-4567","johndoe@website.com","123.123.123","","1000","1234567","1234567","","12345","",""

4. Order Tracking

Persistent Data: Order Number, SubID

Reference Data: History KeyID, Reference KeyID

Definitions:

Order – An order is all transactions of Check Pre-Note, Check Pre-Auth, Check Settlement, Check Return, Check Late Return for an order by a consumer.

Transaction – This is one particular piece of a transaction as in the Check Pre-Note, Check Pre-Auth, etc...

Transaction Block – Is the block of transactions from validating the account / requesting the monies to receiving the monies or receiving a return. One order can have several transaction blocks for the initial transactions and the recurring transactions.

To Determine Initial Transactions:

Take all transactions for a date range then parse them out by Check Pre-Note and Check Pre-Auth where the Reference KeyID is blank and Recurring Description is Initial. This number gives you the total number of initial signups during that time period.

Refunds:

Refunds only occur after a Check Settlement. We cannot refund money that we haven't received yet.

Tracking the Stages of an Order:

Each product sold will have a persistent Order Number throughout the life span of the order, even when it is in a recurring stage. The combination of the Order Number, Reference KeyID and History KeyID will let you track the step-by-step transactions that led to the current status of an order.

The Pre-Note stage will never have a Reference KeyID, but will carry a History KeyID.

Please refer to the Transaction Life Cycle below for clarification on the below paragraph.

Linking up Transactions for a particular order.

If you get a Check Return but you don't know where it occurred in the order or from what transaction block, you can use referencekeyid found with the Check Return entry to start the process of finding which block it came from for this particular order. Take this reference keyid and look for a transaction that contains that referencekeyid as the historykeyid. This should return the Check Pre-Auth for which we received the Check Return. You can use this process for any Check Return, Check Late Return, Check Refund, Check Settlement, ACH NOC, etc... until you find a transaction entry that has a blank referencekeyid. This puts you at the beginning of this particular transaction block.

One thing to note is that each time we recur a transaction the Check Pre-Auth will have a blank referencekeyid indicating that a new transaction block is starting. The easiest way to see the transaction blocks in order would be to grab all transactions for a particular orderid and then sort those by date and referencekeyid, which should put those into order from start to finish. Remember, sometimes you may see a Check Settlement for the recurring billing before the initial trial price settles.

Another Approach if you wanted to display each transaction block would be to grab all Check Pre-Note / Check Pre-Auth sorted by date with a blank referencekeyid. Then, go through each of those using the historykeyid and find the next transaction in the list by looking for the next transaction that contains that history keyid as the referencekeyid. Grab the history keyid for that transaction and look for the transaction that contains that as the referencekeyid. You can continue this process until you get no more results indicating that you have hit the end of the transaction block and continue with the next transaction block in the list.

Retries are an exception. Retry Pre-Auths will have a recurring description of "initial," just like an initial transaction. They are to be viewed as breaks in the transaction process, meaning, we didn't get the money now we are running the account through our virtual collections department in an attempt to retrieve the funds. We will retry an account a total of two more times after the return. If we get the money from our virtual collections department then we will

restart the billing. After that they go into our live collections process in which we send out letters to the consumer asking for them to remit payment. If payment is received we do not restart the billing at this time.

Our recommendation is to import everything because the data can be used to your advantage later. Order Number and SubID will always be persistent, and if you sort the transactions by date, you can get a very clear idea of what went on with the account. In rare cases, we send two transactions per Order Number in one day, but timestamps should show the definitive order.

****Note:** We only Pre-Note new transactions. If we have already processed transactions through this customer's account there is no reason to Pre-Note a second time. We will simply move the transaction to Pre-Auth status and request the funds from the customer's bank account.

ACH Transaction Types

Transaction Type will be defined by one of the following terms:

Check Pre-Note: Means that we are sending a Pre-Notification of the pending debit entry to allow the RDFI (consumer's bank) the opportunity to return (**Check Return**) or correct (**Notice of Change (NOC)**) the item. The pre-notification process involves validation of the ABA and bank account number only.

Check Pre-Auth: This is when we request the funds from the bank (initiate the ACH debit entry).

Check Settlement: This indicates that the **Check Pre-Auth** has not been returned by the RDFI (consumer's bank) prior to the specified time period indicated in the Service Agreement the merchant and WTS entered into

Check Return: This means that the RDFI (consumer's bank) has returned the item prior to the specified time period defining a **Check Settlement** (Settled Funds) indicated in the WTS Service Agreement.

Check Late Return: This means that a **Return** was received by WTS after a **Check Settlement**.

Check Refund: We have refunded this money back to the consumer.

Notice of Change (NOC): This means a response from the RDFI that WTS needs to make a change to the ABA or bank account data in order to properly process the entry.

ACH Transaction Life Cycle

The following diagram outlines the possible life cycle of an ACH transaction:

Check Pre-Note → Check Pre-Auth → Check Settlement
or
Check Pre-Note → Check Pre-Auth → Check Settlement → Check Refund
or
Check Pre-Note → Check Pre-Auth → Check Settlement → Check Late Return
or
Check Pre-Note → Check Pre-Auth → Check Return
or
Check Pre-Note → Check Return
or
Check Pre-Auth → Check Settlement
or
Check Pre-Auth → Check Settlement → Check Refund
or
Check Pre-Auth → Check Settlement → Check Late Return
or
Check Pre-Auth → Check Return

Additional Resources

FAQ's: <http://achbill.com/mercsupport/>