**The SpeedChex HTTP POST API for Merchants**
# ECheck Transaction Commands
**Version 1.0.1**


## Introduction

The ECheck Transaction Commands detailed in this API give merchants a comprehensive platform for authorizing, modifying, voiding and processing electronic check payments through their own software.

The purpose of this document is not only to explain how to submit an ACH payment for authorization and processing, but also to present the concepts necessary for understanding what data is required, why, and what the response data should mean to your software application and your customers.  Please read each section carefully


## The SpeedChex Gateway and the Command/Response System

The SpeedChex Gateway supports processing for multiple payment methods including credit cards, ATM pin-debit cards, electronic checks (ACH), and remote deposit of scanned checks (Check 21).

Software applications can communicate with the SpeedChex Gateway through any one of the following established Internet protocols:

- SOAP 1.1 Web Services
- SOAP 1.2 Web Services w/MTOM attachment support
- Traditional HTTP POST

Although each of these communication protocols requires API documentation that is specific to the functionality of that protocol, each API shares a common command/response method for interacting with the gateway.

Simply put, your software issues a command to the SpeedChex Gateway to accomplish any specific task and the gateway will send back a formatted response to your command.

The Command/Response system is quite extensive and supports the ability to perform a variety of transaction management tasks including:

- Creating/Authorizing new payment transactions
- Uploading batches of transactions
- Modifying/Cancelling existing transactions or batches
- Retrieving reports
- Querying payment data

This document, like all other API documents for the SpeedChex Gateway, is targeted toward a specific subset of commands that are grouped according to either payment method, the task(s) to be performed, or both.

## Available ECheck Transaction Commands

The following is a list and brief explanation of the ECheck Transaction Commands that can be issued through the *SpeedChex HTTP POST API:*

- o **ECheck.Authorize** – Validates the transaction data packet for errors, performs basic routing and account number validation and optionally verifies the bank account using the *Express Verify* service. Does not process the electronic check through the Federal Reserve, or verify or lock funds.

- o **ECheck.Capture** – Schedules a previously authorized ECheck payment for processing through the Federal Reserve at the next End-of-Day Batch Settlement date/time.

- o **ECheck.ProcessPayment** – Processes payments from or payments to a bank account. Performs an *ECheck.Authorize* and *ECheck.Capture* in a single command. Payment is only processed to the Federal Reserve if it passes the authorization process.

- o **ECheck.VerifyBankAccount** – Performs basic algorithm verification on a routing number and account number and optionally verifies the bank account using the *Express Verify* service.

- o **ECheck.Update** – Modifies a transaction if it has not yet been sent to the Federal Reserve for processing at the daily cut-off time.

- o **ECheck.Refund** – Reverses a previous debit that has settled (Cleared) and refunds the same amount (or less if specified) to the same bank account as the original referenced debit. Refunds are automatically captured for settlement.

- o **ECheck.Void** – Cancels (stops) an ECheck payment from processing if it has not yet been sent to the Federal Reserve for processing.

- o **ECheck.Hold** – Places a Scheduled transaction on indefinite 'Merchant Hold' status if it has not yet been sent to the Federal Reserve for processing. Authorized transactions cannot be placed on hold.

- o **ECheck.RemoveHold** – Removes the 'Merchant Hold' status on a transaction and reschedules the transaction for processing on the date specified in the command.

## An Overview of Electronic Check Processing

Electronic checks, also referred to as ACH (Automated Clearing House) transactions, are check transactions processed electronically through the Federal Reserve. Just as with paper checks, merchants may utilize the ACH system both to send and to receive check payments between the merchant and their customers, employees, vendors, etc.

Unlike credit cards, electronic check transactions cannot be instantly verified and approved. This is due to the rules that govern the United States Federal Reserve and the fragmented structure of the current banking system. The banking industry is working to make all forms of check processing and authorization occur in real-time, but due to political, economic, and consumer protection issues, that technology is still years away from widespread implementation.

What has been implemented instead is a system that still provides authorization, but over a period of days instead of seconds. There are several reasons for the slowness of the process, but the bottom line is that the process has been made logistically slow on purpose to protect bank account owners from fraud and error.

The following chart shows a step-by-step breakdown of the authorization process for an electronic check transaction:

**Steps of an Electronic Check Authorization**
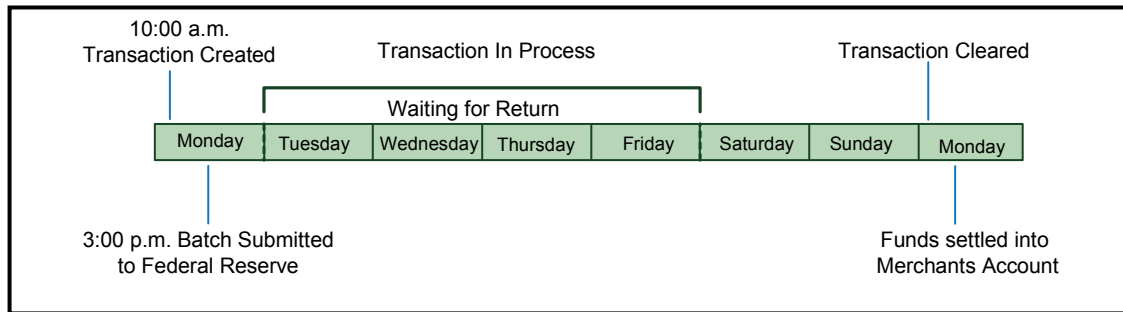


Instead of being sent between banks in real-time, all electronic check transactions are batched and sent between banks overnight through the Federal Reserve (on banking business days only).  A bank can take up to four business days to indicate if a transaction is rejected (Returned) and why.  If no response is received within that timeframe, the transaction is considered approved (Cleared).

Please note that although banks are expected to respond with any Returns within the four business day timeframe, there are some exceptions to this rule because of laws that protect the customer.  If a Return is received after a check payment has been Cleared and funds have been settled to the merchant's account, the Return transaction is classified as a Charged Back because funds will need to be "charged back" from the merchant's account and returned to the customer's bank.

## Data Security and Protection

Protecting the financial transaction data processed through the SpeedChex Gateway is of utmost priority. This means not only implementing the highest levels of security standards in data encryption and system security, but also setting strict controls that limit authorized access to sensitive information.

Every merchants is assigned a unique Merchant ID, GateID, and GateKey that must be kept confidential and will be required as part of each data packet sent to the SpeedChex Gateway.  In addition, an IP filtering scheme may be implemented to ensure that command  packets are only accepted from IP addresses registered by the merchant.

## Verification Using *Express Verify*

Merchants may choose to sign up for an optional bank account verification service called *Express Verify*. This service can report in real-time whether an account exists, or whether it is currently overdrawn, frozen or closed thus ascertaining whether a check is likely to be returned.

The service returns a 3-letter verification status which can be "POS" (positive) indicating the bank account is found and in good standing, "NEG" (negative) indicating the account does not exist or is not in good standing, and "UNK" (unknown) indicating the bank account does not belong to a participating bank.  The code "ERR" (error) can also result if technical problems occurred verifying the account.

Please see ***Appendix B – Express Verify Response Codes*** for a complete list of possible responses from the *Express Verify* system and their meanings.


## Application Testing

Merchants can test the **SpeedChex HTTP POST API** by simply including an optional field called *TestMode* and setting the value of that field to "*On*".  Test commands sent with "*TestMode=On*" will receive valid responses from the CCFS Payment Gateway but the command will not actually be processed by the CCFS system.

The following information may be helpful when testing your application:

> **Test Merchant Gateway Credentials**
> MerchantID:  2001
> Merchant_GateID:  test
> Merchant_GateKey:  test
>
> **Test Bank Account that Passes *Express Verify***
> Routing Number:  123123123
> Account Number:  <any number>
>
> **Test Bank Account that Fails *Express Verify***
> Routing Number:  123123123
> Account Number:  987654321

***Please Note:***  Merchant ID 2001 is a demo account.  Any information you transmit may be viewed by users running the CCFS demo.  This includes payment accounts, names, addresses, phone numbers, and email addressees.  Please always use fabricated test information.


## Unique Payment Transaction Identification

Proper communication between two separate transaction management applications (like this gateway and your software application) requires that both applications share a common, unique reference for each transaction in order for the two applications to communicate intelligently.

The SpeedChex Gateway supports the following methods for uniquely referencing transactions:

> **Transact_ReferenceID**
>
> At the completion of each successful *Echeck.Authorization, ECheck.ProcessPayment* or *ECheck.Refund* command, the *Transact_ReferenceID* value returned by the gateway is the value that must be used to capture, update or void a pending transaction.  The *Transact_ReferenceID* value is also included in all transaction reports as a tool for cross-referencing your internal transactions with the report results.  Your software will need to store this value and associate it internally with your own payment transaction record for later transaction cross-reference.
>
> **Provider_TransactionID**
>
> The SpeedChex Gateway also supports the ability for merchants to assign their own unique internal ID's to each payment transaction.  Although this value cannot be used to capture, update, or void pending transactions, the *Provider_TransactionID* value will be included in all transaction reports as an alternative (and easier) method of cross-referencing your internal transactions with the report results.  It will also be possible to query transactions using *Provider_TransactionID* values.  And finally, during batch uploads, the *Provider_TransactionID* is actually required in order to facilitate deferred response processing.

## Standard Entry Class (SEC) Codes and Payment Authorization

NACHA regulations require that a transaction submitted to the Federal Reserve for processing must include something called a Standard Entry Class (SEC) Code to communicate exactly how the customer gave the merchant authorization to debit/credit their bank account.

The following table shows the proper SEC Codes to use depending on how the merchant obtained authorization to debit/credit an individual or company's bank account:

### Available Standard Entry Class (SEC) Codes

| Authorization Method | SEC Code |
|---|---|
| Document Signed by Individual | PPD |
| Document Signed by Company | CCD |
| Via the Internet | WEB |
| Recorded Telephone Call | TEL |
| Non-Recorded Telephone Call with Notification Sent via First Class Mail | TEL |
| *Paper Check Scanned and Converted to Electronic Check at Point-of-Sale | POP |
| *Paper Check Received via Mail/Courier Scanned and Converted to Electronic Check | ARC |
| *Check Received at Point-of-Sale Scanned and Converted to Electronic Check in Back Office | BOC |

\* Please note that POP, ARC, and BOC transactions can
only be created using the Remote Deposit Batch API

If NACHA decides to audit a merchant or processor, part of the audit process may require providing proof of the authorization method (SEC Code) specified for any given transaction. Failure to properly comply and provide proof of the authorization can result in fines up to $10,000 for each transaction in violation, so it is important that you correctly indicate the SEC Code and maintain good records of your authorizations. Additional information about SEC Codes can be provided upon request.

## Overview of the Command Process

Integrating this API into your software application is not difficult.  The following is an overview of the major components of this task:

- **Data Gathering** - Solution Providers are responsible for collecting and submitting all data associated with a transaction command.

- **Submitting a Gateway Command –** You must create a script or program that submits a command packet over the Internet using the secured HTTPS protocol.  The rules for constructing the HTTPS command are defined in the next section of this document titled *General Implementation Rules and Specifications*.

- **Transaction Packet Validation** – The SpeedChex Payment Gateway parses the command packet for missing data, invalid field values, security violations, etc.

- **Response Processing** - The SpeedChex Payment Gateway will return a comma-delimited, JSON, or XML text response after it receives and processes the command.  The exact format and meaning of the text response will be based on the command issued as defined in the next section of this document titled *General Implementation Rules and Specifications*.

## General Implementation Rules and Specifications

1. **Basic Command Template** – The following table defines the basic data fields that will be required when submitting any command through the ***SpeedChex HTTP POST API:***

| Field Name | Usage | Field Value Format Constraints |
|---|---|---|
| MerchantID | Required | Provided by your Account Manager |
| Merchant_GateID | Required | Provided by your Account Manager |
| Merchant_GateKey | Required | Provided by your Account Manager |
| Command | Required | Set this value to the name of the command you are issuing to the SpeedChex Payment Gateway. |
| CommandVersion | Required | Set to **1.0** to enforce this API documentation revision. |
| ResponseType | Optional | Value must be: **JSON**, **XML**, or **CSV**.  Default is **CSV** |
| TestMode | Optional | Set this value to 'On'  to test a command response without actually executing the command.  Default value is 'Off'. |
| *<additional fields as required>* |  | Based on the Command value, you may be required to define additional fields to send in the command packet.  These fields will be defined in the various sections of this document dedicated to each specific command. |

2. **Submitting a Gateway Command** – To submit a gateway command to SpeedChex, use either HTTP POST or GET to transmit the necessary command fields to the following secure URL:

   https://www.speedchex.com/datalinks/transact.aspx

   If using HTTP GET, please make sure that all data values have been encoded to be URL-safe.

3. **Response Template** – In response to any command, the SpeedChex Payment Gateway will return a simple comma-delimited text response indicating whether the command succeeded or failed and any additional information necessary to interpret the response. The following table defines the field values that will be returned in every response and their meaning:

| Field Name | Field Contents | Max Length | Additional Information |
|---|---|---|---|
| CommandStatus | Returns one of the following values: • *Approved* • *Declined* • *Error* | 30 | Indicates the success or failure of the command issued. |
| ResponseCode | A 3 digit code indicating command success or reason for command failure. | 3 | Please refer to *Appendix A - Response Code Definitions* for a list of possible ResponseCode values, their descriptions, and what ErrorInformation may be made available. |
| Description | Description of the *ResponseCode* value | 255 | |
| ErrorInformation | Additional information to help determine the source of an error. | 50 | |
| *ExpressVerify*.Status | Returns one of the following values if *Express Verify* is activated: 'POS', 'NEG', 'UNK' or 'ERR'. | 3 | ExpressVerify is a complex object that contains the bank account verification results from *Express Verify*. |
| *ExpressVerify*.Code | A code indicating the reason for the *ExpressVerify*.Status value | 5 | Please refer to **Appendix B - Express Verify Response Codes** for more details about these responses. |
| *ExpressVerify*.Description | A brief explanation for the *ExpressVerify*.Status value | 255 | |
| ResponseData | Please see the documentation for the specific command to be issued for an explanation of the possible value(s) for this field. | | This is a generic object that can take the form of any scalar or complex object called for by the command that is issued. |
| Provider_TransactionID | The *Provider_TransactionID* value defined in the command. | 50 | This value can be used as a transaction reference when processing responses must be deferred (like for batch uploads) |
| PaymentKey™ | A unique token assigned to the credit card account just used AND the billing profile associated with that credit card. | 255 | Merchant must sign up for the PaymentKeys™ service. |
| Transact_ReferenceID | A unique ID assigned to each command submitted to the SpeedChex Gateway. | 30 | This value can be used for as a unique transaction identifier or as a reference for support on any command. |

**Transaction Commands –** *ECheck.Authorize*

**Command:** **ECheck.Authorize**

**Description:** Validates the transaction data packet for errors, performs basic routing and account number validation and optionally verifies the bank account using the *Express Verify* service. Does not process the electronic check through the Federal Reserve, or verify or lock funds. The following table defines the data field rules for this command:

| Field Name | Usage | Field Value Format Constraints | Max Length |
|---|---|---|---|
| MerchantID | Required | Provided by your Account Manager | - |
| Merchant_GateID | Required | Provided by your Account Manager | - |
| Merchant_GateKey | Required | Provided by your Account Manager | - |
| Command | Required | Set to *ECheck.Authorize* | 50 |
| CommandVersion | Required | Set to *1.0* for this API documentation revision. | - |
| TestMode | Optional | Set this value to *On* to test a command response. Default value is *Off*. | 3 |
| Provider_TransactionID | Optional | A unique identifier assigned to this transaction | 50 |
| PaymentDirection | Required | Value must be *FromCustomer* or *ToCustomer* | 12 |
| Amount | Required | The amount of the check. | - |
| CheckType | Required | Value must be *Personal* or *Business* | 8 |
| AccountType | Required | Value must be *Checking* or *Savings* | 8 |
| RoutingNumber | Required | ABA routing number on customer's check. | 9 |
| AccountNumber | Required | Customer's bank account number | 30 |
| CheckNumber | Conditional | Customer's check number. Only required for non-ACH transactions. | 25 |
| LocationName | Optional | Any location pre-defined by the merchant in their administration console | 50 |
| Merchant_ReferenceID | Optional | The internal ID or invoice number the merchant wants assigned to this payment | 50 |
| Description | Optional | A description for this transaction | 100 |
| Billing_CustomerID | Optional | A unique ID assigned to the Customer. | 50 |
| Billing_CustomerName | Required | Name of bank account holder | 80 |
| Billing_Company | Conditional | Company name on bank account. Required if *CheckType* field is set to *Business* | 80 |
| Billing_Address1 | Required | Street Address | 70 |
| Billing_Address2 | Optional | Additional street address information | 40 |
| Billing_City | Required | City | 70 |
| Billing_State | Required | 2-letter state abbreviation | 30 |
| Billing_Zip | Required | Zip Code (format: ##### or #####-####) | 10 |
| Billing_Country | Optional | 2-letter country code (ISO 3166). Default is *US* | 2 |
| Billing_Phone | Required | Phone number | 20 |
| Billing_Email | Conditional | Payment notification email address. Required if *SendEmailToCustomer* is set to *Yes* | 80 |
| SendEmailToCustomer | Required | Value must be *Yes* or *No* | 3 |
| Customer_IPAddress | Conditional | Customer's IP Address. Only required if the *SECCode* field is set to *WEB*. | 15 |
| Run_ExpressVerify | Required | Value must be *Yes* or *No* | 3 |
| SECCode | Required | Value must be *PPD*, *CCD*, *WEB*, or *TEL*. | 3 |
| ResponseType | Optional | Must be *JSON*, *XML*, or *CSV*. Default is *CSV* | 4 |

**Transaction Commands –** *ECheck.Capture*

**Command:  ECheck.Capture**

**Description:**  Schedules a previously authorized ECheck payment for processing through the Federal Reserve at the next End-of-Day Batch Settlement date/time.  The following table defines the data field rules for this command:

| Field Name | Usage | Field Value Format Constraints | Max Length |
|---|---|---|---|
| MerchantID | Required | Provided by your Account Manager | - |
| Merchant_GateID | Required | Provided by your Account Manager | - |
| Merchant_GateKey | Required | Provided by your Account Manager | - |
| Command | Required | Set to *ECheck.Capture* | - |
| CommandVersion | Required | Set to *1.0*  for this API documentation revision | - |
| TestMode | Optional | Set this value to *On* to test a command response.  Default value is *Off*. | 3 |
| Transact_ReferenceID | Required | The unique *Transact_ReferenceID* sent in response to the original *ECheck.Authorize* command. | 50 |
| Provider_TransactionID | Optional | Updates the unique identifier the merchant wants assigned to this transaction | 50 |
| DateScheduled | Optional | Date to process payment. (format: mm/dd/yyyy) | - |
| Amount | Optional | Provides option to process a different  payment amount than was originally authorized. | - |
| Merchant_ReferenceID | Optional | Update the internal ID or invoice number the merchant wants assigned to this payment | 50 |
| ResponseType | Optional | Must be: *JSON*, *XML*, or *CSV*.  Default is *CSV* | 4 |

**Transaction Commands – *ECheck.ProcessPayment***

**Command:  ECheck.ProcessPayment**

**Description:** Processes payments from or payments to a bank account.  Performs an *ECheck.Authorize* and *ECheck.Capture* in a single command.  Payment is only processed to the Federal Reserve if it passes the authorization process.  The following table defines the data field rules for this command:

| Field Name | Usage | Field Value Format Constraints | Max Length |
|---|---|---|---|
| MerchantID | Required | Provided by your Account Manager | - |
| Merchant_GateID | Required | Provided by your Account Manager | - |
| Merchant_GateKey | Required | Provided by your Account Manager | - |
| Command | Required | Set to *ECheck.ProcessPayment* | 50 |
| CommandVersion | Required | Set to *1.0* for this API documentation revision. | - |
| TestMode | Optional | Set this value to *On* to test a command response.  Default value is *Off*. | 3 |
| Provider_TransactionID | Optional | A unique identifier assigned to this transaction | 50 |
| DateScheduled | Optional | Date to process payment. (format: mm/dd/yyyy) | - |
| PaymentDirection | Required | Value must be *FromCustomer* or *ToCustomer* | 12 |
| Amount | Required | The amount of the check. | - |
| CheckType | Required | Value must be *Personal* or *Business* | 8 |
| AccountType | Required | Value must be *Checking* or *Savings* | 8 |
| RoutingNumber | Required | ABA routing number on customer's check. | 9 |
| AccountNumber | Required | Customer's bank account number | 30 |
| CheckNumber | Conditional | Customer's check number.  Only required for non-ACH merchants. | 25 |
| LocationName | Optional | Any location pre-defined by the merchant in their administration console | 50 |
| Merchant_ReferenceID | Optional | The internal ID or invoice number the merchant wants assigned to this payment | 50 |
| Description | Optional | A description for this transaction | 100 |
| Billing_CustomerID | Optional | A unique ID assigned to the Customer. | 20 |
| Billing_CustomerName | Required | Name of person on bank account | 80 |
| Billing_Company | Conditional | Company name on bank account.   Required if *CheckType* field is set to *Business* | 80 |
| Billing_Address1 | Required | Street Address on bank account | 70 |
| Billing_Address2 | Optional | Additional street address information | 40 |
| Billing_City | Required | City | 70 |
| Billing_State | Required | State | 30 |
| Billing_Zip | Required | Zip Code (format: ##### or #####-####) | 10 |
| Billing_Country | Optional | 2-letter country code (ISO 3166).  Default is *US* | 2 |
| Billing_Phone | Required | Phone number | 20 |
| Billing_Email | Conditional | Payment notification email address.  Required if *SendEmailToCustomer* is set to *Yes* | 80 |
| SendEmailToCustomer | Required | Value must be *Yes* or *No* | 3 |
| Customer_IPAddress | Conditional | Customer's IP Address.  Only required if the *SECCode* field is set to *WEB*. | 15 |
| Run_ExpressVerify | Required | Value must be *Yes* or *No* | 3 |
| SECCode | Required | Value must be *PPD*, *CCD*, *WEB*, or *TEL*. | 3 |
| ResponseType | Optional | Must be: *JSON*, *XML*, or *CSV*.  Default is *CSV* | 4 |

**Transaction Commands – *ECheck.VerifyBankAccount***

**Command:** **ECheck.VerifyBankAccount**

**Description:** Performs basic algorithm verification on a routing number and account number and optionally verifies the bank account using the *Express Verify* service. The following table defines the data field rules for this command:

| Field Name | Usage | Field Value Format Constraints | Max Length |
|---|---|---|---|
| MerchantID | Required | Provided by your Account Manager | - |
| Merchant_GateID | Required | Provided by your Account Manager | - |
| Merchant_GateKey | Required | Provided by your Account Manager | - |
| Command | Required | Set to *ECheck.VerifyBankAccount* | 50 |
| CommandVersion | Required | Set to *1.0* for this API documentation revision. | - |
| TestMode | Optional | Set this value to *On* to test a command response. Default value is *Off*. | 3 |
| PaymentDirection | Required | Value must be *FromCustomer* or *ToCustomer* | 12 |
| Amount | Required | The amount of the check. | - |
| CheckType | Required | Value must be *Personal* or *Business* | 8 |
| AccountType | Required | Value must be *Checking* or *Savings* | 8 |
| RoutingNumber | Required | ABA routing number on customer's check. | 9 |
| AccountNumber | Required | Customer's bank account number | 30 |
| Run_ExpressVerify | Required | Value must be *Yes* or *No* | 3 |
| ResponseType | Optional | Must be: *JSON*, *XML*, or *CSV*. Default is *CSV* | 4 |

**Response Note:** If *Run_ExpressVerify* is set to **Yes**, the SpeedChex Gateway will return a failure *ResponseCode* value of **202** (**Failed Express Verify**) when the *ExpressVerify.Status* value returns is **NEG** (negative).

If the *ExpressVerify.Status* value is **POS** (positive) or **UNK** (unknown), the *ResponseCode* value will be set to **000** (**Command Successful.  Approved.**).

Please refer to **Appendix B - Express Verify Response Codes** for more details about the possible responses from the *Express Verify* service to make sure your code handles each type of response appropriately according to your company's or your merchant's policies.

**Transaction Commands – *ECheck.Update***

**Command:  ECheck.Update**

**Description:** Modifies a transaction if it has not yet been sent to the Federal Reserve for processing. The following table defines the data field rules for this command:

**Note:**  Please do not specify a value for an optional field if you do not intend to modify its value.

| Field Name | Usage | Field Value Format Constraints | Max Length |
|---|---|---|---|
| MerchantID | Required | Provided by your Account Manager | - |
| Merchant_GateID | Required | Provided by your Account Manager | - |
| Merchant_GateKey | Required | Provided by your Account Manager | - |
| Command | Required | Set to *ECheck.Update* | 50 |
| CommandVersion | Required | Set to *1.0*  for this API documentation revision. | - |
| TestMode | Optional | Set this value to *On* to test a command response.  Default value is *Off*. | 3 |
| Transact_ReferenceID | Required | The unique *Transact_ReferenceID* sent in response to the original *ECheck.Authorize* or *ECheck.ProcessPayment* command. | 50 |
| Provider_TransactionID | Optional | A unique identifier assigned to this payment | 50 |
| DateScheduled | Optional | Date to process payment. (format: mm/dd/yyyy) | - |
| PaymentDirection | Optional | Value must be *FromCustomer* or *ToCustomer* | 12 |
| Amount | Optional | The amount of the check. | - |
| CheckType | Optional | Value must be *Personal* or *Business* | 8 |
| AccountType | Optional | Value must be *Checking* or *Savings* | 8 |
| RoutingNumber | Optional | ABA routing number on customer's check. | 9 |
| AccountNumber | Optional | Customer's bank account number | 30 |
| CheckNumber | Optional | Customer's check number.  Only required for non-ACH transactions. | 25 |
| LocationName | Optional | Any location pre-defined by the merchant in their administration console | 50 |
| Merchant_ReferenceID | Optional | The internal ID or invoice number the merchant wants assigned to this payment | 50 |
| Description | Optional | A description for this transaction | 100 |
| Billing_CustomerID | Optional | A unique ID assigned to the Customer. | 20 |
| Billing_CustomerName | Required | Name of person on bank account | 80 |
| Billing_Company | Conditional | Company name on bank account.   Required if *CheckType* field is set to *Business* | 80 |
| Billing_Address1 | Required | Street Address on bank account | 70 |
| Billing_Address2 | Optional | Additional street address information | 40 |
| Billing_City | Required | City | 70 |
| Billing_State | Required | State | 30 |
| Billing_Zip | Required | Zip Code (format: ##### or #####-####) | 10 |
| Billing_Country | Optional | 2-letter country code (ISO 3166).  Default is *US* | 2 |
| Billing_Phone | Required | Phone number | 20 |
| Billing_Email | Conditional | Payment notification email address.  Required if *SendEmailToCustomer* is set to *Yes* | 80 |
| SendEmailToCustomer | Optional | Value must be *Yes* or *No* | 3 |
| Customer_IPAddress | Optional | Payer's IP Address.  Only required if the *SECCode* field is set to *WEB*. | 15 |
| Run_ExpressVerify | Optional | Value must be *Yes* or *No* | 3 |
| SECCode | Optional | Value must be *PPD*, *CCD*, *WEB*, or *TEL*. | 3 |
| ResponseType | Optional | Must be: *JSON*, *XML*, or *CSV*.  Default is *CSV* | 4 |

**Transaction Commands – *ECheck.Refund***

**Command: ECheck.Refund**

**Description:** Reverses a previous debit that has settled (Cleared) and refunds the same amount (or less if specified) to the same bank account as the original referenced debit. Refunds are automatically captured for settlement. The following table defines the data field rules for this command:

| Field Name | Usage | Field Value Format Constraints | Max Length |
|---|---|---|---|
| MerchantID | Required | Provided by your Account Manager | - |
| Merchant_GateID | Required | Provided by your Account Manager | - |
| Merchant_GateKey | Required | Provided by your Account Manager | - |
| Command | Required | Set to *ECheck.Refund* | - |
| CommandVersion | Required | Set to *1.0* for this API documentation revision | - |
| TestMode | Optional | Set this value to *On* to test a command response. Default value is *Off*. | 3 |
| Transact_ReferenceID | Required | The unique *Transact_ReferenceID* sent in response to the original debit *ECheck.Authorize* or *ECheck.ProcessPayment* command. | 50 |
| Provider_TransactionID | Optional | A unique identifier the merchant wants assigned to the new refund payment | 50 |
| DateScheduled | Optional | Date to process this payment. | - |
| Amount | Optional | Use this field if you need to specify a refund amount less than the original debit amount. | - |
| Merchant_ReferenceID | Optional | The internal ID or invoice number the merchant wants assigned to this refund transaction. | 50 |
| Description | Optional | A description for the refund transaction | 100 |
| ResponseType | Optional | Must be: *JSON*, *XML*, or *CSV*. Default is *CSV* | 4 |

**Transaction Commands – *ECheck.Void***

**Command:  ECheck.Void**

**Description:**  Cancels (stops) a transaction from processing to the Federal Reserve.  The following table defines the data field rules for this command:

| Field Name | Usage | Field Value Format Constraints | Max Length |
|---|---|---|---|
| MerchantID | Required | Provided by your Account Manager | - |
| Merchant_GateID | Required | Provided by your Account Manager | - |
| Merchant_GateKey | Required | Provided by your Account Manager | - |
| Command | Required | Set to ***ECheck.Void*** | 50 |
| CommandVersion | Required | Set to ***1.0***  for this API documentation revision. | - |
| TestMode | Optional | Set this value to ***On*** to test a command response.  Default value is ***Off***. | 3 |
| Transact_ReferenceID | Required | The unique *Transact_ReferenceID* sent in response to the original *ECheck.Authorize* or *ECheck.ProcessPayment* command. | 50 |
| ResponseType | Optional | Must be: ***JSON***, ***XML***, or ***CSV***.  Default is ***CSV*** | 4 |

**Transaction Commands – *ECheck.Hold***

---

**Command:  ECheck.Hold**

**Description:**  Places a scheduled transaction on 'Merchant Hold' if it has not yet been sent to the Federal Reserve for processing.  This is a delay in processing that a merchant may impose on a transaction until they are ready to remove the hold.  Authorized transactions cannot be placed on hold.

The following table defines the data field rules for this command:

| Field Name | Usage | Field Value Format Constraints | Max Length |
|---|---|---|---|
| MerchantID | Required | Provided by your Account Manager | - |
| Merchant_GateID | Required | Provided by your Account Manager | - |
| Merchant_GateKey | Required | Provided by your Account Manager | - |
| Command | Required | Set to *ECheck.Hold* | 50 |
| CommandVersion | Required | Set to *1.0*  for this API documentation revision. | - |
| TestMode | Optional | Set this value to *On* to test a command response.  Default value is *Off*. | 3 |
| Transact_ReferenceID | Required | The unique *Transact_ReferenceID* sent in response to the original *ECheck.Authorize*, *ECheck.ProcessPayment* or *ECheck.Refund* command. | 50 |
| ResponseType | Optional | Must be: *JSON*, *XML*, or *CSV*.  Default is *CSV* | 4 |

**Transaction Commands – *ECheck.RemoveHold***

<br>

**Command: ECheck.RemoveHold**

**Description:** Removes the 'Merchant Hold' status on a transaction and reschedules the transaction for processing on the date specified in the command.  The following table defines the data field rules for this command:

| Field Name | Usage | Field Value Format Constraints | Max Length |
|---|---|---|---|
| MerchantID | Required | Provided by your Account Manager | - |
| Merchant_GateID | Required | Provided by your Account Manager | - |
| Merchant_GateKey | Required | Provided by your Account Manager | - |
| Command | Required | Set to **ECheck.RemoveHold** | 50 |
| CommandVersion | Required | Set to **1.0**  for this API documentation revision. | - |
| TestMode | Optional | Set this value to **On** to test a command response.  Default value is **Off**. | 3 |
| Transact_ReferenceID | Required | The unique *Transact_ReferenceID* sent in response to the original *ECheck.Authorize, ECheck.ProcessPayment* or *ECheck.Refund* command. | 50 |
| DateScheduled | Required | Date to process this payment. | - |
| ResponseType | Optional | Must be: **JSON**, **XML**, or **CSV**.  Default is **CSV** | 4 |

## Appendix A – Response Code Definitions

| Response Code | Description | Contents of the *ErrorInformation* Field |
|---|---|---|
| **GATEWAY COMMAND SUCCESS** | | |
| 000 | Command Successful.  Approved. | |
| | | |
| **GATEWAY COMMAND ERRORS** | | |
| 100 | Invalid Gateway Credentials | Credential Object Name |
| 101 | Invalid Gateway Command | |
| 102 | Duplicate Command Not Processed | Transact_ReferenceID of the original Command |
| 103 | Transaction Cannot Be Modified | Transaction Status |
| 104 | Batch Cannot Be Modified | Batch Status |
| 105 | Invalid Transact_ReferenceID | |
| 106 | Invalid BatchID | |
| 107 | Non-Unique Reference/Transaction ID | Field Name |
| 108 | Invalid Reference/Transaction ID | Field Name |
| 109 | Invalid Source IP | |
| 110 | Invalid Value In Message | |
| | | |
| **INPUT DATA VALIDATION ERRORS** | | |
| 150 | Required Field Missing | Field Name |
| 151 | Field Value Is Not Valid | Field Name |
| 152 | Field Value Exceeds Maximum Length | Field Name |
| | | |
| **PAYMENT ACCOUNT VERIFICATION FAILURES** | | |
| 200 | Failed AVS | |
| 201 | Failed CVN | |
| 202 | Failed Express Verify | |
| 203 | Invalid Credit Card Number | |
| 204 | No Such Card Issuer | |
| 205 | Expired Card | |
| 206 | Invalid Expiration Date | |
| 208 | Call Issuer for Further Information | |
| 209 | Invalid Routing Number | |
| 210 | Invalid Bank Account Number | |
| 211 | Invalid PIN | |
| 212 | Invalid PaymentKey | |
| | | |
| **PAYMENT ACCOUNT DECLINES** | | |
| 300 | Transaction was Declined by Processor | |
| 301 | Transaction was Rejected by Gateway | |
| 302 | No Card Number on File with Issuer | |
| 304 | Invalid Account Type | |
| 305 | Account Closed | |
| 306 | Account Inactive | |

*Response Code Definitions Continued on Next Page...*

## Appendix A – Response Code Definitions

| Response Code | Description | Contents of the ErrorInformation Field |
|---|---|---|
| **PAYMENT ACCOUNT DECLINES (*continued...*)** | | |
| 307 | Account Frozen | |
| 309 | Insufficient Funds | |
| 310 | Over Limit | |
| 311 | Do Not Honor | |
| 312 | Transaction Not Allowed | Reason (if known) |
| 313 | Invalid for Debit | Reason (if known) |
| 314 | Invalid for Credit | Reason (if known) |
| 315 | Customer Opt Out | Reason (if known) |
| 316 | Customer Advises Not Authorized | |
| 317 | Manual Key Not Allowed | |
| 318 | Duplicate Transaction at Processor | |
| 319 | PaymentKey Authentication Failed | |
| | | |
| **FRAUD DECLINES** | | |
| 400 | Pick Up Card | |
| 401 | Lost Card | |
| 402 | Stolen Card | |
| 403 | Fraudulent Card | |
| 404 | Excessive Declines From Same Source | |
| 405 | Excessive PIN Attempts | |
| 406 | Excessive Purchase Frequency | |
| | | |
| **MERCHANT DIRECTIVES FROM PROCESSOR** | | |
| 500 | Declined - Stop All Recurring Payments | |
| 501 | Declined - Update Cardholder Data Available | |
| 502 | Declined - Further Instructions Available | Instructions |
| 503 | Declined - Call Processor for Voice Authorization | |
| 504 | Declined - Call Processor for Fraud Instructions | |
| | | |
| **PROCESSOR ADMINISTRATIVE ERRORS** | | |
| 600 | Internal Gateway Error | |
| 601 | Internal Processor Error | |
| 602 | Communication Error with Issuer | |
| 603 | Communication Error with Processor | |
| 604 | Processor Feature Not Available | |
| 605 | Processor Format Error | |
| 606 | Invalid Terminal Number | |
| 607 | Merchant Not Setup | |
| 608 | Merchant Account is Inactive | |
| 609 | Invalid Merchant Configuration | |
| 610 | Invalid Payment Method for Merchant | |
| 611 | Unsupported Card Type | |
| | | |
| **OTHER** | | |
| 999 | Contact Support Representative | |

## Appendix B – Express Verify Response Codes

Bank accounts that are found and in good standing are approved with a POS (positive) result status. Accounts that do not exist or are in a negative standing at their bank are rejected with a NEG (negative) result status. If a bank account is from a bank that is not part of the *Express Verify* network or the bank is not reporting information about the bank account, the transaction is approved with an UNK (unknown) result status.

The following table shows all possible responses from the *Express Verify* system:

| Status | Code | Description |
|--------|------|-------------|
| ERR | E01 | EXPRESS VERIFY SERVICE NOT ACTIVATED |
| NEG | P00 | ACCT NOT LOCATED |
| NEG | P01 | ACCOUNT CLOSED |
| NEG | P03 | NO DEBITS |
| NEG | P04 | NO CHECKS |
| NEG | P05 | NSF |
| NEG | P06 | UNCOLLECTED FUNDS |
| NEG | P12 | ISSUER DECLINED |
| UNK | P40 | NO INFO |
| NEG | P41 | NEGATIVE INFO |
| UNK | P50 | NON PARTICIPANT |
| POS | P70 | VALIDATED |
| POS | P72 | VALIDATED AMOUNT |
| NEG | V02 | ACCOUNT NOT APPROVED |
| NEG | V10 | INVALID ROUTING NUMBER |
| UNK | V90 | PREAUTH VENDOR UNAVAILABLE |
| UNK | V91 | PREAUTH VENDOR ERROR |