Transaction History API

Table of Contents

This document details the transaction history Application Programming Interface (API), a new addition to the E*TRADE API. It is an addendum to the E*TRADE Financial API Reference Guide [https://content.etrade.com/etrade/estation/pdf/API_Technical_Documentation.pdf].

1. Introduction

The transaction history APIs allow users to retrieve customer account transactions. Typically users of these APIs will be individual developers or third party vendors who are building software applications to sell to E*TRADE customers.

Applications can use the E*TRADE transaction history API to retrieve all transactions for a given account, all transactions of a specific type, or transactions within categories such as deposits, withdrawals, trades, currency exchange, sweep, and corporate actions.

2. Transaction API

The Accounts module of the E*TRADE API allows users to review specific account information such as balances and current positions. The `transactions` service is a new resource within the Accounts module that permits the retrieval of transaction history. Refer to the Accounts section in Chapter 4 of the E*TRADE Financial API Reference Guide [https://content.etrade.com/etrade/estation/pdf/API_Technical_Documentation.pdf] for information about the other Accounts resources.

2.1. Transactions Resource

The following table contains information about the `transactions` resource.

Table 1. Transactions Resource

Item	Detail
Description	Return transaction history details for a specified account
URL	https://etws.etrade.com/accounts/rest/*AccountId*/transactions
Sandbox URL	https://etwssandbox.etrade.com/accounts/sandbox/rest/*AccountId*/transactions
HTTP Method	GET

2.2. Request URLs

A request URL has the following required path elements.

```
/accounts/rest/AccountId/transactions
```

These elements may be followed by optional inputs that take the form of additional path elements, or by query parameters, which follow the question mark (“?”) symbol. *AccountId* is the only required input and appears as part of the required path. Refer to the Transaction History Query Syntax section to review the syntax rules for path elements and parameters.

The default response is XML response data. JSON response data may be selected by appending `.json` to the end of the path elements in the request URL. This option may be used regardless of other path elements present.

The following table lists the path elements.

Table 2. Path Elements

Property Name	Default Value
AccountId	Eight-digit account number assigned to the user
Group	Major transaction group; refer to the table in the *Group*s section for a list of groups
AssetType	Asset type within the TRADES group; refer to the table in the *AssetType* section for a list of asset types
TransactionType	Transaction type within the group; refer to the table in the *TransactionType* section for a list of transaction types

Property Name	Default Value
TickerSymbol	TRADES transactions may be filtered by ticker symbol
JsonFlag	Selects JSON response rather than XML, if present

The following table lists the parameters.

Table 3. Parameters

Field	Data Type	Description
`fromDate`	string	MMDDYYYY format
`toDate`	string	MMDDYYYY format
`count`	integer	Returns record count
`marker`	integer	Not for use by developer. Refer to the note in the Basic Output section for more information

2.3. *Groups*

Transaction types can be viewed in broad categories which are referred to as groups. These groups are listed in the table below. They represent all possible transaction types for a customer's account. API developers can use these categories to retrieve all of the historical transactions for a particular group; for example, to retrieve all deposits the customer made during a particular time period.

Table 4. Groups

Field	Data Type	Description
`deposits`	string	Transactions related to cash deposits in the brokerage account
`withdrawals`	string	Transactions related to cash withdrawals in the brokerage account
`trades`	string	Transactions related to trading, such as dividends payments, trade events, expiration events, and assignment events

2.4. *AssetType* (TRADES Group Only)

The *AssetType* may be selected only if the TRADES group is specified in the request. The API supports further grouping of TRADES transactions for Equities, Options, Money Market Funds, Mutual Funds, and Bonds. Specifying the *AssetType* is optional and additional path refinements may also be used regardless of whether or not it is specified.

Table 5. AssetType

Field	Data Type	Description
`eq`	string	Equities
`optn`	string	Options
`mmf`	string	Money Market Funds
`mf`	string	Mutual Funds

Field	Data Type	Description
bond	string	Bonds

2.5. *TransactionType*

In addition to providing developers the ability to retrieve transactions by a predefined *Group* or for the TRADES group, by a predefined *AssetType*, the API allows retrieval by a specific transaction type or by transaction type(s) within a group. Note that when a *Group* has been specified, *TransactionType* filtering is limited to types appropriate to the group as indicated by the Group column in the following table.

Table 6. Transaction Types (D=Deposits, W=Withdrawals, T=Trades)

Field	Group	Description
assignment	T	The receipt of an exercise notice by an options writer that requires him or her to sell (call option) or to buy (put option) the security at the specified strike price. Also referred to as exercise assignment or exercise notice. Also refers to the transfer of ownership of an asset from one person to another.
atm	D, W	Service charge from automated teller machines
check	D, W	Check deposit or withdrawal
corporate_actions		Split, merger, or acquisition
contribution	D, W	Money put into retirement fund
currency_xch		Currency exchange
debit	W	List of transactions that decrease assets
deposit	D	List of transactions that Increase assets
direct_debit	W	Decrease of assets
direct_deposit	D	Increase of assets
distribution	D, W	Money taken out of retirement fund
dividend	D	Dividend paid
exercise	T	To implement the rights of an option, by buying (for call options) or selling (for put options) the underlying asset. Also, to exchange a right or a warrant for the appropriate amount of stock.
expiration	T	The date on which an option, right, or warrant expires and becomes worthless if not exercised. Also, the date on which an agreement is no longer in effect.
fee	W	Service fees
interest	D	Interest charged
pose	D, W	Point of sale debit
sweep		Sweep deposit
transfer	D, W	Cash transfer; can be into or out of the account
wire	D, W	Cash wire; can be transaction decrease or increase

2.6. Responses

There are two forms of response data returned by the API: basic output and transaction details output. The following subsections describe the fields that are part of each of these outputs.

2.6.1. Basic Output

Basic output is returned in a transactions list. This list contains a `count` of the number of transaction items to follow. If the number of records retrieved exceeds the requested `count`, then the `next` field will also be present. This field can be used to retrieve the subsequent transaction set(s).

Table 7. `transactions` Basic Output

Field	Data Type	Description
`count`	integer	Number of transactions returned
`next`	string	URL to fetch next set of records
`transaction list`	list<Transaction>	List of transaction items

Each transaction consists of the items shown below.

Table 8. Basic Output for Each Transaction

Field	Data Type	Description
`transactionDate`	long	Date is in Unix time format
`description`	string	One-line description
`amount`	decimal	Dollars and cents or 0.00 if not applicable
`transactionShortDesc`	string	Short description of action; for example, `Sold`
`transactionId`	long	Unique ID of transaction
`details`	string	URL to use to retrieve the transaction details

Note

If the `next` field is present, the URL will contain a `marker` parameter. The `marker` should not be modified when retrieving additional transactions and it should not be set in the initial request.

2.6.2. Transaction Details Output

The following table shows the fields returned in a transaction details response.

Table 9. Transaction Details Response Output

Field	Data Type	Description
`transactionDate`	long	Date of the specified transaction
`transactionType`	string	Type of transaction
`userDescription`	string	User-defined description

Field	Data Type	Description
transactionDescription	string	Transaction description long name
quantity	decimal	Share count
amount	decimal	Dollars and cents or 0.00 if not applicable
commission	decimal	Commission amount
price	decimal	Price paid
productID		Instrument description
productID.symbol	string	Instrument symbol
productID.typeCode	string	Instrument type: equity, bond, mutual fund, index
category	string	User-defined category
displaySymbol	string	Client-facing symbol
underlyingProductId		Underlying instrument description
underlyingProductId.symbol	string	Underlying instrument symbol
underlyingProductId.typeCode	string	Underlying instrument type: equity, bond, mutual fund, index
settlementDate	long	Settlement date
settlementCurrency	string	Settlement currency
paymentCurrency	string	Payment currency
accountOrderNo	long	Order number

2.7. Transaction History Query Syntax

This section describes the query syntax for the transaction history API. Inputs, shown in italics, are replaced with values as described in the rules below.

Note

Path elements and parameters are not case sensitive.

2.7.1. URL Syntax

Following is the syntax of a transaction history API request.

```
https://etws.etrade.com
/accounts/rest/AccountId/transactions/Group/TransactionTypes?Parameters
```

1. The required path elements are shown in plain face. Italic items are optional except for *AccountId*. A *JsonFlag* (not shown) can be appended to the path to select the alternate response format. See the Request URLs section for details.

2. These are the optional items:

   - *Group* - refer to the *Group*s section for a list of valid group names

   - *TransactionTypes* is a comma-separated list of *TransactionType*. See the *TransactionType* section for a list of valid transaction types. If *TransactionTypes* is specified, a *Group* must also be present.

- *Parameters* is an ampersand-separated list of query parameters in standard URL format delineated by "?". Refer to the Request URLs section for the list of valid parameters.

2.7.2. Exception to the Syntax: TRADES Group

The following syntax is specific to the TRADES group. Italic items are optional except for *AccountId*.

```
https://etws.etrade.com/accounts/rest/AccountId/transactions
/TRADES/AssetType|ALL|TransactionTypes/Symbol?Parameters
```

1. *AssetType* is optional, but its use is only permitted with the TRADES group. It will limit retrieved trades to the selected asset class. Refer to the *AssetType* section for a list of asset types.
2. ALL is only available with the TRADES transaction group
3. *TransactionTypes* is as defined in the URL Syntax section
4. *Symbol* is a stock symbol (for example, GOOG)

2.7.3. Exception to the Syntax: Filtering by a Single Transaction

It is possible to filter by a single transaction type by using the following syntax.

```
https://etws.etrade.com/accounts/rest/AccountId/transactions
/TransactionType?Parameters
```

Refer to the *TransactionType* section for the list of transaction types.

2.7.4. Available History

The transaction history API supports two rolling years of transaction history.

2.7.5. API Parameter Defaults

These are the defaults for the parameters:

1. `fromDate` - if `fromDate` is not specified, 30 business days of history prior to the `toDate` will be provided
2. `toDate` - if `toDate` is not specified, the current date will be used
3. `count` - the default record count is 50
4. `marker` - a `marker` will be provided in the `next` response field for requests that result in more than the requested record count. See the note in the Basic Output section for more information.

3. Sample API Responses

This section shows sample XML and JSON responses that result from typical requests. As described in the Request URLs section, the JSON response can be obtained by appending the *JsonFlag* to the request path. The transaction details response can be obtained by issuing a subsequent REST call using the `details` URL string returned in the basic response data.

3.1. XML Basic Details

Request:

```
https://etws.etrade.com/accounts/rest/61055940/transactions
```

Response:

```
<transactions>
  <count>3</count>
  <transaction>
    <transactionId>120321003898030</transactionId>
    <transactionDate>1328083200</transactionDate>
    <transactionShortDesc>Adjustment</transactionShortDesc>
    <description>FROM 11/26 THRU 12/25 @ 8.140%BAL 31 AVBAL 31 MARGIN INTEREST
 CORRECTION</description>
    <amount>-0.21</amount>
    <details>https://etws.etrade.com/accounts/rest/61055940/
transactions/120321003898030</details>
  </transaction>
  <transaction>
    <transactionId>120321003898020</transactionId>
    <transactionDate>1328083200</transactionDate>
    <transactionShortDesc>Adjustment</transactionShortDesc>
    <description>FROM 11/26 THRU 12/25 @ 8.440%BAL 31 AVBAL 31 REVERSE MARGIN
 INTEREST</description>
    <amount>0.22</amount>
    <details>https://etws.etrade.com/accounts/rest/61055940/
transactions/120321003898020</details>
  </transaction>
  <transaction>
    <transactionId>120261031120990</transactionId>
    <transactionDate>1327564800</transactionDate>
    <transactionShortDesc>Interest</transactionShortDesc>
    <description>FROM 12/26 THRU 01/25 @ 8.440%BAL 32 AVBAL 32</description>
    <amount>-0.23</amount>
    <details>https://etws.etrade.com/accounts/rest/61055940/
transactions/120261031120990</details>
  </transaction>
</transactions>
```

3.2. JSON Basic Details

Request:

```
https://etws.etrade.com/accounts/rest/61055940/transactions.json
```

Response:

```
{"json.transactions":{"count":3,"transactionList":[{"transactionId":120321003898030,
"transactionDate":1328083200,"transactionShortDesc":"Adjustment","description":
"FROM 11\/26 THRU 12\/25 @ 8.140%BAL 31 AVBAL 31 MARGIN INTEREST CORRECTION",
"amount":-0.21,"details":
"https:\/\/etws.etrade.com\/accounts\/rest\/61055940\/transactions\/120321003898030"},
```

{"transactionId":120321003898020,"transactionDate":1328083200,"transactionShortDesc":
"Adjustment","description":
"FROM 11\/26 THRU 12\/25 @ 8.440%BAL 31 AVBAL 31 REVERSE MARGIN INTEREST",
"amount":0.22,"details":
"https:\/\/etws.etrade.com\/accounts\/rest\/61055940\/transactions\/120321003898020"},
{"transactionId":120261031120990,"transactionDate":1327564800,"transactionShortDesc":
"Interest","description":"FROM 12\/26 THRU 01\/25 @ 8.440%BAL 32 AVBAL 32",
"amount":-0.23,"details":
"https:\/\/etws.etrade.com\/accounts\/rest\/61055940\/transactions
\/120261031120990"}]}}

3.3. XML Transaction Details

Request:

```
https://etws.etrade.com/accounts/rest/61055940/transactions/120321003898020
```

Response:

```
<transactionDetails>
  <transactionDate>1328083200</transactionDate>
  <transactionType>Adjustment</transactionType>
  <userDescription></userDescription>
  <transactionDescription>FROM 11/26 THRU 12/25 @ 8.440%BAL 31 AVBAL 31 REVERSE MARGIN
 INTEREST</transactionDescription>
  <quantity>0</quantity>
  <amount>0.22</amount>
  <price>0</price>
  <commission>0</commission>
  <productId>
    <symbol></symbol>
    <typeCode>UNKNOWN</typeCode>
  </productId>
  <underlyingProductId/>
  <displaySymbol></displaySymbol>
  <accountOrderNo>0</accountOrderNo>
  <settlementCurrency>USD</settlementCurrency>
  <paymentCurrency>USD</paymentCurrency>
  <category></category>
  <settlementDate>1328083200</settlementDate>
</transactionDetails>
```

3.4. JSON Transaction Details

Request:

```
https://etws.etrade.com/accounts/rest/61055940/transactions/120321003898020.json
```

Response:

{"json.transactionDetails":{"transactionDate":1328083200,"transactionType":
"Adjustment","userDescription":"","transactionDescription":
"FROM 11\/26 THRU 12\/25 @ 8.440%BAL 31 AVBAL 31 REVERSE MARGIN INTEREST",
"quantity":0,"amount":0.22,"price":0,"commission":0,
"productId":{"symbol":"","typeCode":"UNKNOWN"},"underlyingProductId":"",
"accountOrderNo":0,"settlementCurrency":"USD","paymentCurrency":"USD","category":"",
"settlementDate":1328083200}}

3.5. XML Format Error

Response:

```
<Error>
  <ErrorCode>1000</ErrorCode>
  <ErrorMessage>Invalid AccountId. It is either missing or has invalid value.
  AccountId can have up to 8 digits.</ErrorMessage>
</Error>
```

3.6. JSON Format Error

Response:

```
{"Error":{"errorCode":1000,"message":"Invalid AccountId. It is either missing or has
invalid value. AccountId can have up to 8 digits."}}
```

4. Use Cases

This section covers possible API use cases. The host portion of the path (`https://etws.etrade.com`) must be prepended to each of these examples.

4.1. Retrieve All Transactions

This is the default use case.

```
/accounts/rest/AccountId/transactions
```

1. All transaction types will be received
2. The record count will default to 50 since it was unspecified by the `count` parameter in the URL
3. The default time range will be 30 days prior to today (`fromDate`) to today (`toDate`)

4.2. Retrieve Transactions Using Filtering

In addition to retrieving all transactions, API developers can use filters to retrieve transactions for a specific group, retrieve transactions for a particular transaction type, and perform other retrievals of the customer's transaction history. The following use cases illustrate this.

1. Retrieve `TRADES` transactions by date

   ```
   /accounts/rest/AccountId/transactions/TRADES?&startDate=MMDDYYYY&endDate=MMDDYYYY
   ```

   a. All trade-related transactions made during the specified date range will be received

   b. The record count will default to 50 since it was unspecified by the `count` parameter in the URL

2. Retrieve a certain number of transactions by specifying the `count`

   ```
   /accounts/rest/AccountId/transactions/TRADES?count=30
   ```

   a. Only trade transactions will be received since the `TRADES` group is specified

   b. Since a record count of 30 is specified, one set of 30 records based on the most recent 30 trades will be received

   c. The default time range will be 30 days prior to today (`fromDate`) to today (`toDate`)

3. Retrieve transactions from the `DEPOSITS` group

   ```
   /accounts/rest/AccountId/transactions/DEPOSITS
   ```

   a. All deposits will be received

   b. The record count will default to 50 since it was unspecified by the `count` parameter in the URL

   c. The default time range will be 30 days prior to today (`fromDate`) to today (`toDate`)

4. Retrieve transactions for the TRADES group by asset type

   ```
   /accounts/rest/AccountId/transactions/TRADES/EQ
   ```

a. All equity trades will be received

b. The record count will default to 50 since it was unspecified by the `count` parameter in the URL

c. The default time range will be 30 days prior to today (`fromDate`) to today (`toDate`)

5. Retrieve transactions by ticker symbol

```
/accounts/rest/AccountId/transactions/TRADES/EQ/GOOG
```

a. All equity trades for ticker symbol GOOG will be received

b. The record count will default to 50 since it was unspecified by the `count` parameter in the URL

c. The default time range will be 30 days prior to today (`fromDate`) to today (`toDate`)

6. Retrieve transactions by ticker symbol without regard for asset type

```
/accounts/rest/AccountId/transactions/TRADES/ALL/GOOG
```

a. All equity and derivative trades for ticker symbol GOOG will be received

b. The record count will default to 50 since it was unspecified by the `count` parameter in the URL

c. The default time range will be 30 days prior to today (`fromDate`) to today (`toDate`)

4.3. Retrieve Transactions for Multiple Transaction Types

It is possible to retrieve transaction history for multiple transaction types by specifying a comma-separated list of transaction type items. This is illustrated with the following example.

```
/accounts/rest/AccountId/transactions/TRADES/ASSIGNMENT,EXPIRATION
```

1. All assignments and expirations for the TRADES group will be received
2. The record count will default to 50 since it was unspecified by the `count` parameter in the URL
3. The default time range will be 30 days prior to today (`fromDate`) to today (`toDate`)

5. Error Messages

The following table lists the various error messages that may be returned. It is an addendum to *Table 1: Account Services Error Messages* in Appendix A of the E*TRADE Financial API Reference Guide [https://content.etrade.com/etrade/estation/pdf/API_Technical_Documentation.pdf].

Table 10. Account Services Error Messages

Error Code	Error Message
1000	Invalid *AccountId*. Missing or invalid value. *AccountId* can have up to eight digits.
1023	Invalid date range. Both `fromDate` and `toDate` should be provided; `toDate` should be greater than `fromDate` and `fromDate` should fall within the last two years.
1024	Invalid *TransactionType*. *TransactionType* is not supported for the specified group.
1025	Invalid *Group*.
1026	*AssetType* is not supported for the specified transaction group or multiple *AssetType*s were provided, which is not allowed.
1027	Invalid `count` value; value should be between 1 and 50.