



E*TRADE Developer Platform

Developer Guide and API Reference

Contents

Getting Started	5
Introduction.....	6
Architecture	6
Authorization.....	6
Agreements	7
Support & Community.....	8
The E*TRADE API.....	9
About the API.....	9
API Syntax.....	10
API Documentation.....	11
Terminology.....	12
Next Steps.....	13
Developer Guide.....	14
Authorizing the Application	15
Overview.....	15
The User Experience	17
Using OAuth Credentials	19
The OAuth API Module	21
The Sandbox Environment	22
Overview.....	22
Accessing the Sandbox	22
About Rate Limits	24
Overview.....	24
Limits	24
Requests for Multiple Items.....	24
The Get Rate Limits API	25
About Notifications.....	26
Alerts	26
Notifications	26
Streaming Updates	26
The Streaming API	27
Introduction.....	27
Notifications	27
Using Comet.....	29
Sample code.....	30
REST API Reference	34
Authorization	35
Get Request Token.....	35
Authorize Application	37
Get Access Token	39
Renew Access Token	41
Revoke Access Token	43
Accounts	45

List Accounts	45
Get Account Balance	48
Get Account Positions.....	53
List Alerts.....	62
Read Alert.....	68
Delete Alert.....	71
Get Transaction History	73
Get Transaction Details	87
Market	91
Look Up Product.....	91
Get Quote	94
Get Option Chains	115
Get Option Expire Dates	121
Orders	124
List Orders	124
Preview Equity Order	142
Place Equity Order.....	149
Preview Equity Order Change.....	157
Place Equity Order Change	163
Preview Option Order	170
Place Option Order	178
Preview Option Order Change	187
Place Option Order Change.....	194
Cancel Order	202
Rate Limits	205
Get Rate Limits	205
Notifications.....	208
Get Notifications	208
Error Handling	212
Overview.....	212
HTTP Status Codes	214
Account Module Error Messages	215
Market Module Error Messages	216
Order Module Error Messages.....	217
Notification Error Messages.....	220
Code Resources	221
Tutorial: Start Developing in Java	222
Introduction.....	222
Requirements	222
Step 1: OAuth Authorization.....	222
Step 2: Retrieve a list of accounts.....	224
What's next?	225
E*TRADE SDK Guides	226
Using the Java SDK.....	226
Using the PHP SDK.....	229
Using the VC++ SDK	238
Code Snippets: Accounts	259

Initialize Accounts Client	259
List Accounts	259
Get Account Balance	259
Get Account Positions.....	260
List Alerts.....	260
Read Alert.....	260
Delete Alert.....	261
Code Snippets: Market	262
Initialize Market Client.....	262
Get Quote	262
Look Up Product.....	263
Get Option Expire Dates	263
Get Option Chains	263
Code Snippets: Order.....	265
Initialize Order Client.....	265
List Orders	265
Preview Equity Order	266
Place Equity Order.....	266
Preview Equity Order Change.....	267
Place Equity Order Change	268
Preview Option Order	269
Place Option Order	270
Preview Option Order Change	271
Place Option Order Change.....	272
Cancel Order	273

Getting Started

This section provides a brief overview of the E*TRADE Developer Platform and offers suggestions on how to start learning and using the API.

Introduction

The E*TRADE Developer Platform enables E*TRADE customers and developers to create their own investment applications that leverage E*TRADE's extensive market data offerings, order-routing capabilities, and other services.

The platform's API also allows E*TRADE customers who currently use a third-party trading platform to view E*TRADE account and market information and place trade orders directly to E*TRADE from that platform.

Using the E*TRADE Developer Platform, a client application can:

- Authenticate E*TRADE customers
- Directly manage trading: place orders, modify or cancel orders, and check order status
- Review specific information for an account, such as balances and current positions
- Retrieve price and other information about an index, stock, or option
- Receive system messages from E*TRADE

Architecture

The E*TRADE Developer Platform provides most of its services via a REST API. Most of the API features are accessed via simple HTTP GET requests. Requests that require detailed input, such as an order to buy or sell stock, use an HTTP POST request, with the parameters included as either XML or JSON data. The HTTP DELETE request is also used, with one API, to delete messages after they have been read.

Although most of the platform uses a RESTful request-response model, a streaming API is also provided, so that an application can receive timely "push" notifications of order status changes.

Authorization

The E*TRADE REST API uses the OAuth protocol to authorize every service request. In practice, this means that your application must enable users to log in to their E*TRADE account and click a consent button to grant access for each session. For your application to do this, even for testing purposes, requires an OAuth consumer key - a code that uniquely identifies the application (i.e., the service consumer) and its developer.

Consumer keys

We support two levels of consumer key. An *individual* key is tied to a single user ID, and allows access for only that user. This is appropriate for developing applications for personal use, or for the early stages of a larger development effort. A *vendor* key, on the other hand, permits access by multiple users and is appropriate for applications that may be widely distributed.

Note that separate keys are required for accessing "sandbox" data (used for development and testing purposes) versus actual production data. For this reason, a typical developer has at least two consumer keys.

Requesting keys

To request a key, you must have an E*TRADE account. (You will also need the account so that you can log in and use your application.) It can be a personal, professional, or corporate account. If you don't have one, you can quickly set one up online at <http://www.etrade.com>. For development purposes, it is not necessary to fund the account, but to do any actual trading will require funds.

When you are logged in to your account, request a consumer key via a secure message on this page: <https://us.etrade.com/e/t/accounts/sccreatemsg>. Select the subject "Technical Issue" and the topic "E*TRADE API". Be sure to:

- State that you are requesting a consumer key
- Explain the type of application you are developing
- Specify whether you want an individual or vendor key

You may also want to request that your user account be approved as a non-funded account, so that you're not required to have funds in the account to keep it open.

Within a few business days, a sandbox consumer key will be sent back to you by secure message, along with a Developer Agreement which you will need to review and sign. When we've received your agreement and processed your request, we'll issue you a production consumer key.

When you get the key, you will be notified of its scheduled expiration date - typically, two years from its creation date.

Note that if you are assigned an individual key, rather than a vendor key, it will only work with your E*TRADE account. Attempting to use an individual key with a different user account will result in an error.

Agreements

Developers and users are asked to sign several types of agreements:

- **API Agreement** - As a developer, you are asked to sign an API agreement before you will be issued a consumer key for production data. As mentioned earlier, requesting a key requires that you log in to your E*TRADE account and send a secure message.
- **Authorization** - As a user, whenever the application starts a session with the E*TRADE platform, you must log in to your E*TRADE account and agree to an authorization request.

- **RTQ Agreement** - As a user, you must sign the Real-Time Quote Agreement before you can access data with the `quote` API. You can do this at <http://www.etrade.com>.
- **Extended-hours Agreement** - As a user, you must sign the Extended-Hours Disclosure Agreement before you can perform after-hours trading (if the application supports this feature). You can do this at <http://www.etrade.com>.

Support & Community

The [E*TRADE Developer Platform discussion board](#) is a rich resource where you can learn more about the platform and share ideas and information with the E*TRADE developer community. You may find answers there to technical problems or design challenges. Whether you're a longtime E*TRADE developer or just getting started, we encourage you to participate in this active development forum.

The E*TRADE API

The E*TRADE Developer Platform provides a REST API and related resources for creating customized investing applications, opening the door not just to trading applications but to new applications and hybrids for algorithmic trading, social investing, and market intelligence. The API can be used by individual E*TRADE investors building their own custom solutions as well as third-party vendors building solutions for any E*TRADE customer.

About the API

Most of the platform services are REST APIs that provide information such as account lists, quotes, and alerts, or functionality such as the ability to submit trade orders. Since the API is RESTful, it is easy to integrate into your application statelessly, without complex session management, and it flexibly delivers data in either JSON or XML format.

In addition, the platform supports a streaming API that lets you subscribe to timely "push" notifications of order status updates.

API Versions

As the E*TRADE API evolves through multiple versions, development of new features always takes place on the most current version of the API. Previous versions of the API are not revised except in the case of critical bug fixes.

The original platform version is designated as version 0 or "v0".

API Modules

The APIs are organized into modules as shown in the table below. The module names are used as part of the API syntax, and the same modules are used to organize the API documentation.

API Module	Description	APIs
OAuth	Acquire, renew, and revoke OAuth token(s).	Get Request Token, Authorize Application, Get Access Token, Renew Access Token, Revoke Access Token
Accounts	Get information on the user's accounts, including transaction histories, as well as any alerts for the user (trade executions, order expirations, bill payments, etc.).	List Accounts, Get Account Balance, Get Account Positions, List Alerts, Read Alert, Delete Alert, Get Transaction History, Get Transaction Detail
Market	Look up symbols and company information, get quotes, fundamentals, and performance history, and build option tables.	Get Option Chains, Get Option Expiration Dates, Look Up Product, Get Quote

Order	Get a list of current orders, preview/place/modify/cancel orders for equities and options.	List Orders, Cancel Order, Preview Equity Order, Place Equity Order, Preview Equity Order Change, Place Equity Order Change, Preview Option Order, Place Option Order, Preview Option Order Change, Place Option Order Change
Streaming API	Subscribe to receive timely push notifications of order status changes - fulfillment, cancellation, etc.	<i>See documentation for details on streaming subscriptions.</i>
Notifications	Get notifications about the system and platform, such as new features.	Get Notifications
Rate Limits	Check the API rate limits and see how many requests are still available in the current time interval.	Get Rate Limits

API Syntax

In version v0, most of the REST API calls are GETs, some are POSTs, and one call uses a DELETE (to delete a message). In most of these, the URL has very similar syntax. Here's an example:

```
https://etws.etrade.com/market/rest/quote/MSFT.json?detailFlag=FUNDAMENTAL
```

After the host (*etws.etrade.com*) comes the API module (e.g., *oauth*, *account*, *market*, or *order*), a standard path element */rest/*, and the API endpoint (*quote* in this example). There may be path parameters (*MSFT* in this example). At the end of the path, ".json" may be added to request JSON output instead of the default XML. Finally, some APIs use query parameters, such as the *detailFlag* parameter shown here.

Case sensitivity

The API path and endpoint (*market/rest/quote*) are case-sensitive and should be lower case, as should the ".json" parameter. Path parameters (*MSFT*) and query parameters are not case-sensitive.

Sandbox syntax

The URL path for the sandbox environment is slightly different. Details can be found in our Sandbox documentation.

Data formats

APIs that use HTTP POST will accept data in either XML or JSON format.

Responses use XML format by default. To receive JSON instead of XML, add ".json" in lower case at the end of the path in the request URL.

API Documentation

Our API documentation is organized into groups based on API modules. Within the modules, each API is detailed separately. The following information is typically provided:

- Description
- URL
- HTTP method(s)
- Request parameters, response properties
- Sample requests and responses
- Notes on usage
- Sample use cases
- Related APIs

In-depth topics such as authorization, the sandbox test environment, system messages, and rate limits are discussed separately in Developer Guide articles.

Request parameters

In this documentation, request parameters are shown in tables that indicated whether each parameter is *required* (always present), *conditional* (present under certain circumstances), or *optional* (decided by the developer). For example:

Parameter	Type	Required?	Description
accountId	path	required	Numeric account ID

If parameters must contain one of a specific set of values, the table lists the possible values and the default value (i.e., the value that is assumed if the parameter is not specified).

Response properties

Responses are XML or JSON structures, and some properties of the response may be *complex*, meaning that they are parents of sub-properties. In tables that describe these complex structures, sub-properties are shown indented under the parent, as shown below.

Property	Type	Description
count	integer	Number of transactions in this response
transaction	complex	Container for the elements of a transaction
transactionId	long	Numeric transaction ID
transactionDate	long	Date and time in epoch time

Terminology

This documentation assumes that the developer is familiar with trading concepts and market terminology. For clarification of market terms, you may find the [E*TRADE Financial Help Center](#) helpful.

Dates and times are all US Eastern Time. Most times and dates in API responses are represented as epoch time, i.e., the number of seconds since 12:00am on January 1, 1970.

Next Steps

1. Apply for a consumer key

Follow the instructions [in the introduction](#) to get a consumer key so you can access the platform and run the tutorial. You should receive a sandbox key and Developer Agreement within a few business days.

2. Review the documentation

We recommend starting with the developer guides for authorization and the sandbox. Then familiarize yourself with the APIs. And no matter what language you're using, you may appreciate the simple Java tutorial, which demonstrates getting authorized and requesting account data.

3. Explore the other Developer Platform resources

Our website includes SDKs and sample code for Java, PHP, and VC++. We've posted information on how to partner with us and links to 3rd-party applications that use the E*TRADE API. Finally, you may find valuable ideas and information at our News page or our online [Developer Community](#).

Developer Guide

This section contains guide chapters that introduce specific topics related to the platform:

- [Authorizing the Application](#)
- [The Sandbox Environment](#)
- [Rate Limits](#)
- [About Notifications](#)
- [The Streaming API](#)

Authorizing the Application

Overview

The E*TRADE Developer Platform uses the OAuth authorization protocol, version 1.0a. OAuth enables an authenticated user to authorize limited access to their account by third-party applications, without exposing user credentials or other sensitive information.

This document provides a brief summary of OAuth and describes how it is used in our developer platform. We recommend that developers be familiar with the detailed OAuth information at: <http://oauth.net/core/1.0a/>.

OAuth workflow

The three actors in OAuth are:

Role	Definition	Example
Service provider	A service provider that uses OAuth to let a 3rd-party application have limited access to a user's account	E*TRADE
User	An individual who has an account with the service provider	E*TRADE user who uses your application
Consumer	A website or application that uses OAuth to access the service provider with the user's permission	Your application

A core principle of OAuth is that the service provider can authenticate both the user and the consumer. This provides a secure basis for the user to authorize the consumer for limited access to the user's account on the service provider.

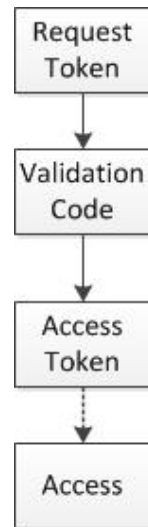
In principle, OAuth authorization requires three steps:

1. The service provider authenticates the consumer.
2. The service provider authenticates the user.
3. The user authorizes the consumer for limited access to the user's account on the service provider.

E*TRADE OAuth lifecycle

With the E*TRADE Developer Platform, the process works like this:

1. The application uses its own credentials to acquire a temporary request token from E*TRADE by calling the **Get Request Token API**.
2. Using the **Authorize Application API**, the application redirects the user, along with the request token, to E*TRADE. There the user logs in to E*TRADE and grants the application limited access to the user's account. E*TRADE generates a verification code, which is passed to the application (manually by the user, or automatically via a callback).
3. The application uses the verification code to acquire an access token that grants temporary access to that user's account. This is done with the **Get Access Token API**.
4. The access token is included with all requests to the E*TRADE API, identifying the user and authorizing the application.



By default, the token expires after two hours of inactivity. At that time, the token may be renewed with the **Renew Access Token API**. When the application terminates or is finished with the token, we recommend that you revoke the token with the **Revoke Access Token API**.

Unless otherwise arranged with customer service, production tokens permanently expire at midnight US Eastern time. A revoked or permanently expired token cannot be renewed. Instead, a new token must be obtained by starting the OAuth lifecycle again as described above.

Callbacks

As mentioned above, when the user authorizes the application, the E*TRADE website generates a verification code that must be passed to the application. One approach is for the user to simply copy the code and paste it into the application. A much better solution is for E*TRADE to automatically redirect the user back to the application, using a *callback URL* with the verification code added as a query parameter, as shown in these example URLs:

```
https://myapplicationsite.com/mytradingapp?oauth_verifier=WXYZ89  
https://myapplicationsite.com?myapp=trading&oauth_verifier=WXYZ89
```

Configuring a callback

Using a callback requires that the callback URL be associated with your consumer key in the E*TRADE system. To request this, log in to your E*TRADE account and send a secure message to Customer Service. Select the subject "Technical Issues" and the topic "E*TRADE API". State that you would like to have a callback configured, and specify your consumer key and the desired callback URL. Your callback URL can be just a simple address, or can also include query parameters.

Once the callback is configured, two system behaviors are changed:

1. The `oauth_callback_confirmed` property of the Request Token API returns TRUE to show that there is a callback URL associated with the consumer key.
2. Users who approve the authorization request are automatically redirected to the callback URL, with the verification code appended as a query parameter (as shown in the example URLs above).

The User Experience

At runtime, the application redirects the user to the E*TRADE site to authorize the application. There, the user authenticates on a webpage that includes a login form similar to this:




The image shows a screenshot of the E*TRADE website's login page. At the top is the E*TRADE logo. Below it is a box titled "SECURE LOG ON" with a padlock icon. Inside the box are two input fields: "User ID:" and "Password:". Below these fields is a green "LOG ON" button with a padlock icon. At the bottom of the box is a link that says "Forgot your User ID or Password?". Below the box is a link that says "New User? Set Up Online Account Access".

Once logged in, the user is presented with a request to authorize the application to access the account, as shown below.

(Note that an *individual* consumer key only works with its original user. If any other user attempts to use that key, a non-specific error message is displayed instead of an authorization form, and the process halts.)



 **Indicate Terms Agreement**

To continue, please acknowledge that you agree to allow the indicated platform to submit requests and retrieve data from your E*TRADE FINANCIAL account(s).

[Vendor Platform Name] will be able to:


- Submit and review orders
- Retrieve account information
- Retrieve market data

[Review the E*TRADE API User Agreement](#)

The name of the application is displayed (based on the consumer key that was passed to the login page) along with the list of privileges that will be granted - e.g., submit and review orders, retrieve account information, and retrieve market data. If the user agrees to the request, E*TRADE generates a verification code that refers to this agreement.

If a callback URL is associated with the consumer key, the browser is automatically redirected to that URL, with the verification key included as a URL parameter. If not, the user sees a page that displays the verification code, as shown below, and the user has to manually copy the code and paste it into a field in the application.



 **Complete Authorization**

Copy the following text code by clicking the "Copy Code" button or selecting it with your cursor and pressing "Control+C" (PC) or "Command+C" (Mac).

You will then be directed to the next page where you will paste the code.

[Review the risks and limitations of the Application Program Interface.](#)

At that point, the application can send a request to E*TRADE for an access token, attaching the consumer key, the verification code, and an appropriate signature based on the application's consumer secret. The access token grants limited access to the user account for a fixed period of time, and must be attached to all API requests as described below.

Using OAuth Credentials

Once the application has acquired OAuth credentials, as described above, they must be attached to all API requests.

OAuth Credentials

Here are the credentials:

Property	Type	Required?	Description
oauth_consumer_key	string	required	The value used by the consumer to identify itself to the service provider.
oauth_timestamp	integer	required	The date and time of the request, in epoch time. Must be accurate within five minutes.
oauth_nonce	string	required	A nonce, as described in OAuth 1.0a documentation - roughly, an arbitrary or random value that cannot be used again with the same timestamp.
oauth_signature_method	string	required	The signature method used by the consumer to sign the request. The only supported option is HMAC-SHA1.
oauth_signature	string	required	Signature generated with the shared secret and token secret using the specified <code>oauth_signature_method</code> , as described in OAuth documentation.
oauth_token	string	required	The consumer's access token issued by the service provider.

Timestamp

A *timestamp* must accompany every REST request, stating the date and time of the request in epoch time (i.e., the number of seconds since 12:00:00 a.m. January 1, 1970 UTC). When the request is received at E*TRADE, the timestamp must be within five minutes of the current UTC time.

Nonce

Every request must include a *nonce* - an arbitrary or random value, used to ensure that each request is unique. The same nonce can be used for multiple requests, as long as those requests do not have the same timestamp. A typical solution is to generate a random hash for each request.

Signature

The application must include a signature as specified in the OAuth documentation at <http://oauth.net/core/1.0a/>. The only supported OAuth version is 1.0a, and the only supported hash method is HMAC-SHA1. For compatibility purposes, we recommend using the libraries provided at <http://oauth.net>.

Note that most of the OAuth examples in this documentation are visually correct but mathematically invalid - i.e., they may not contain values that can be used for testing a signature algorithm. If you wish to test your signature algorithm, use the following table. Given the provided input values, your code should produce the same resulting signature.

Item	Value
Key	c5bb4dcb7bd6826c7c4340df3f791188
Secret	7d30246211192cda43ede3abd9b393b9
Access Token	VbiNYl63EejlKdQM6FeENzcnrLACrZ2JYD6NQROfVI=
Access Secret	XCF9RzyQr4UEPlOa+WIC06BnTfYC1P0Fwr3GUw/B0Es=
Timestamp	1344885636
Nonce	0bba225a40d1bbac2430aa0c6163ce44
HTTP Method	GET
URL	https://etws.etrade.com/accounts/rest/accountlist
Resulting signature	%2FXiv96DzZabnUG2bzPZIH2RARHM%3D

Based on this information, your requests should contain the values shown below, although the variables may be in a different sequence.

HTTP header info

```
Authorization: OAuth
oauth_nonce="0bba225a40d1bbac2430aa0c6163ce44",oauth_timestamp="1344885636",oauth_consumer_key="c5bb4dcb7bd6826c7c4340df3f791188",oauth_token="VbiNYl63EejlKdQM6FeENzcnrLACrZ2JYD6NQROfVI%3D",oauth_signature="%2FXiv96DzZabnUG2bzPZIH2RARHM%3D",oauth_signature_method="HMAC-SHA1"
```

GET request

```
https://etws.etrade.com/accounts/rest/accountlist?oauth_consumer_key=c5bb4dcb7bd6826c7c4340df3f791188&oauth_nonce=0bba225a40d1bbac2430aa0c6163ce44&oauth_signature=%2FXiv96DzZabnUG2bzPZIH2RARHM%3D&oauth_signature_method=HMAC-SHA1&oauth_timestamp=1344885636&oauth_token=VbiNYl63EejlKdQM6FeENzcnrLACrZ2JYD6NQROfVI%3D
```

Submitting credentials

We recommend that you attach the OAuth information to a request by including it in the HTTP header as XML or JSON, with the header name "Authorization" and the value "OAuth". Here is an example:

```
Authorization: OAuth
oauth_consumer_key="282683cc9e4b8fc81dea6bc687d46758",oauth_timestamp="1273254425",
oauth_nonce="LTg2ODUzOTQ5MTEzMTY3MzQwMzE%3D",oauth_signature_method="HMAC-SHA1",
oauth_signature="FjoSQaFDKEDK1FJaz1Y3xArNflk%3D",oauth_token="FiQRgQCRGPo7Xdk6G8QDSEzX0Jsy6sKN14cULcDavAGgU"
```

As an alternative, the OAuth information can be included as URL parameters. This is usually less desirable, because it creates a long query string and tends to mix API parameters with OAuth parameters. Below is an example of such a query string, for a query that would otherwise have been very simple.

```
https://etws.etrade.com/market/rest/quote/IBM?oauth_consumer_key=d85df5b910e3cb47b2a4cf26c20229ae&oauth_nonce=2518627fd47688ae96fec0b2d11b8c24&oauth_signature=FXZWW1N5UkFNcayfpoGXWHsgYNc%3D&oauth_signature_method=HMAC-SHA1&oauth_timestamp=1343164357&oauth_token=eu4zvG1xpZX1A0wGZx4oJDMDPdnn8IsVcVMqNxeKyQ%3D
```

Note that the E*TRADE API does not recognize OAuth credentials in the body of the HTTP request. The only supported options are the header or the query string.

The OAuth API Module

The OAuth module of the E*TRADE Developer Platform includes the following REST APIs:

API	Description
Get Request Token	Returns a request token that can be used to initiate authorization.
Authorize Application	Using the request token, redirects the user to the E*TRADE authorization page, where the user authorizes the consumer application to access the account. This returns a verification code.
Get Access Token	Using the verification code, requests an access token that can be used to access the E*TRADE API on the user's behalf
Renew Access Token	Renews the access token when it expires.
Revoke Access Token	Revokes the access token at the end of the session.

Each of these APIs is documented separately in detail.

The Sandbox Environment

Overview

The E*TRADE Developer Platform provides a sandbox environment where developers can safely experiment with the REST API and perform limited testing of applications, without executing any actual transactions in real markets involving real securities or money.

Here's how it works:

- No actual transactions are executed.
- If you submit a request that contains a syntax error, you receive an appropriate error message in response.
- If you submit a valid request, you receive a sample response from stored data.

Sandbox responses use stored data that's intended to provide typical responses for basic use cases. So the responses you receive will not contain current data, and may not exactly match your requests in other ways. For instance, you might request quotes on GOOG and MSFT, and receive AAPL, ORCL, and CSCO instead.

The sandbox provides a simple way for you to check your syntax, test your deserialization code, see how data appears on your UI, and so on.

Sandbox Samples

Extensive samples of sandbox requests and responses are included in this documentation for reference purposes.

Accessing the Sandbox

Sandbox login

To access the sandbox, you need an OAuth consumer key and secret for the sandbox. OAuth is explained separately under Authorization, and the process of requesting a consumer key is explained in the Getting Started guide.

To "log in" to the sandbox (i.e., to acquire an OAuth access token and secret for the session), use your sandbox consumer key and secret, but otherwise use the same OAuth procedure that you would normally use for actual production data, and the same authorization server.

The access token and secret that you receive in response to this request will only work on the sandbox environment, not in the production environment. So as long as you are using your sandbox credentials, there is no chance of accidentally triggering an actual market transaction, even if you accidentally send your request to a production URL.

Sandbox URLs

Calls to the sandbox use a slightly different syntax than calls to production. The table below shows both. The sandbox syntax is different in two ways - the name of the server is different, and "/sandbox/" is included in the path after the module.

Environment	URL
Production	<code>https://etws.etrade.com/{module}/rest/{API}</code>
Sandbox	<code>https://etwssandbox.etrade.com/{module}/sandbox/rest/{API}</code>

Sample URLs

```
https://etws.etrade.com/accounts/rest/alerts/1108  
https://etwssandbox.etrade.com/accounts/sandbox/rest/alerts/1108
```

About Rate Limits

Overview

The E*TRADE API uses rate limits to manage the volume of incoming requests.

Rate limits are implemented on the Accounts, Market, and Order API modules. The limit is specified at two levels:

- A **"throttling" limit** defines the maximum number of requests that will be processed per second. This is designed to encourage a smooth level of activity and avoid large bursts of requests.
- A **"consumption" limit** defines the maximum requests per hour.

Requests that exceed the limit are not processed. Instead of the expected XML or JSON response, they return an HTML error page containing the message: " Number of requests exceeded the rate limit set ", with the HTTP status code 400. (See the separate documentation on error codes for more information.)

The Get Rate Limits API allows applications to query the limits for the current user and get current usage status. This allows the application to pace itself as needed and avoid failed requests or periods of forced inactivity.

Limits

Rate limits are enforced separately for each consumer key, and within that, separately for each API module. If your application requires higher limits, please contact Customer Service.

The table below shows the limits that are assigned to a typical consumer key. To give a sense of how these limits work in actual practice, the table lists both the maximum requests per hour (the typical tracking interval), and how that breaks down into average requests per second.

Module	Throttling Limit (requests per second per user)	Consumption Limit (requests per hour per user)
Accounts	2	7000
Market	4	14000
Order	2	7000

Requests for Multiple Items

A single request for multiple data items, such as a call to the Get Quote API that includes 25 different stock symbols as parameters, is treated as a single request.

The Get Rate Limits API

For more information on how rate limits may affect your application and how your application can respond, refer to the documentation on the [Get Rate Limits API](#).

About Notifications

The E*TRADE Developer Platform provides three types of communications from the system (in addition to HTTP status messages and system error messages).

Alerts

Alert messages are triggered by various account activity or changes in account status. On the E*TRADE website, you can see examples of alerts listed in a box on the Complete View page, under the Accounts tab. The alerts may involve trade executions, warnings, margin calls outstanding, order expirations, account balances, bill payments, direct deposits, debits, and so on.

In the E*TRADE REST API, the Accounts module provides APIs that can list, display, and delete alerts. A typical application might check for alerts at the start of a session and occasionally throughout the session, especially when displaying an appropriate page - for instance, a landing page or account status page.

For more information, see the documentation for the Accounts module, which describes the List Alerts, Read Alert, and Delete Alert APIs.

Notifications

The Notifications module provides a Get Notifications API that provides system notifications about the E*TRADE service and developer platform that may be of interest to developers. A typical user application would not use this API - it is provided as an information source for developers.

These developer notifications are categorized as either "global" messages that are sent to all developers, or "platform" messages that are targeted at a specific developer or developers. The volume of these kinds of notifications is low.

For more information, see the documentation on the Get Notifications API in the Accounts module.

Streaming Updates

The platform provides a streaming API that pushes timely, low-latency messages out to clients to notify the user of status changes on individual orders - full or partial execution, cancellation, and so on. This streaming API is a separate service, not part of the REST API. To use this service, the client must subscribe to the appropriate channels and "listen" to updates for specified accounts.

Because these order-status updates are time-sensitive and may involve a high volume, a typical interactive application might display them in a status bar, box, or popup. Alternatively, an application might forward these notifications to the user as emails, instant messages, or SMS text messages.

The Streaming API

Introduction

In addition to the REST APIs, the E*TRADE Developer Platform provides a streaming data service that can push data out to an application, rather than requiring the application to poll for the data. This streaming data service is based on the Comet web-service architecture, which uses delayed responses or very long responses to break out of HTTP's single-page, request-response model.

The application begins a session by submitting an HTTP request to the streaming data service. The server sends back a response (containing one or more notifications) when triggered by relevant events, such as a completed order. The application issues fresh HTTP requests as needed to keep the session going.

In this way the application receives low-latency data, without any special message protocols, in a way that does not involve a lot of network traffic or processing.

E*TRADE uses this architecture to push notifications of changes in order status.

Notifications

Notifications are requested by the application on a per-account basis. Notifications are sent only for orders placed under the specified accounts.

A notification about a change in order status contains information about the specific order only; it does not contain information about other orders for the account. If an application requests notifications for an account, and places five orders using that account, then when those orders are executed at least five separate notifications are triggered.

Notification events

Notifications are triggered by changes to order status, including the following:

Event	Description
Partial completion	Order has been partially executed. The status is still live, but the quantity executed has changed.
Completion	Order has been filled.
Cancellation	Order has been cancelled.
Rejection	Order has been rejected.

Notification details

Item	Description
------	-------------

Account Number	Numeric account ID
Order Number	Sequential order number, unique within account
Order Status	Order status: open, pending, executed, cancelled, cancel-requested, etc.
Symbol	Security symbol
Order	Possible values are: <ul style="list-style-type: none"> • BUY • SELL • SHORT • COVER • BUY_TO_OPEN • SELL_TO_CLOSE • SELL_TO_OPEN • BUY_TO_CLOSE
Quantity	Quantity specified in the order. For stocks, this is a number of shares. For mutual funds, this is a dollar value or, if selling the entire position, "All I Own".
Quantity Filled	Quantity that has been executed.
Price	Possible values are: <ul style="list-style-type: none"> • Market • Market_On_Close • Limit • Stop • Stop_Limit
Price	Price for price type
Event Type	See event types listed below.

Sample Notification

Below is a sample order update message.

```
<class>com.etrade.lmq.orderstatus.common.dto.OrderStatusResponse</class>
<quantityFilled>0</quantityFilled>
<accountNumber>83531504</accountNumber>
<price>0</price>
<symbol>AAPL</symbol>
<legStatus>OPEN</legStatus>
<orderNumber>130</orderNumber>
<priceType>TRIGGER_UNSPECIFIED</priceType>
<orderAction>BUY</orderAction>
<quantity>5</quantity>
<eventType>OPEN</eventType>
<orderStatus>OPEN</orderStatus>
```

Complex orders

The following rules describe how complex orders are treated by the streaming service.

- Each leg of an order is pushed as the update is returned from the market. The legs can be pushed separately. Applications can correlate legs by using the `orderNumber` included in the update.
- For Buy Writes, the order status update shows both legs of the order. For partial executions, the quantity filled is pushed for both the stock and the option.
- For Spreads, the order status update shows both legs of the order. For partial executions, the quantity filled is pushed for both legs.

Trailing stops and conditional orders

For trailing stops and conditional orders, an order status is sent if those orders are triggered and executed. If a trailing stop is triggered, an order is sent to the market to execute (triggered orders are market orders), and the notification alerts the user once the order is executed (whether fully or partially).

When orders are cancelled (for example, "One-Cancels-All" orders), or when a user cancels a conditional order, the order is cancelled within the E*TRADE system and not at the exchange. Since the cancellation is a result of user action, no update message is sent.

Note that "Live" and "Open" are not valid statuses. Push events only occur when there is an execution (full or partial), cancellation, or rejection from the exchange.

Using Comet

Our Comet web architecture uses the Bayeux protocol as described in Comet documentation. There are significant differences between Comet 1.x and Comet 2.x. In the E*TRADE API v0, Comet 1.x is supported.

We use the CometD implementation, and to use the service you'll need to use the CometD libraries, available as shown here:

Resource	Link
CometD libraries for JavaScript and Java	http://cometd.org/documentation
CometD documentation for JavaScript	http://cometd.org/documentation/cometd-javascript
CometD documentation for Java	http://cometd.org/documentation/cometd-java

Comet service

The Comet service of the E*TRADE Developer Platform is accessible at this URL:

<https://etwspush.etrade.com/apistream/cometd/oauth/>

This service requires OAuth authorization, which is detailed in a separate section of the Developer Platform documentation. Requests to the service must include an OAuth header -

i.e., the consumer key, access token, timestamp, nonce, signature method, and signature. If the OAuth token expires during the session (e.g., at midnight US Eastern time), subsequent HTTP requests from the application return authorization errors, not streaming updates, so updates effectively cease with the first update after the token expires - even though the subscription to the service is technically still in effect.

The general approach is to connect to the Comet service, subscribe to the update and error channels, specify which accounts to listen to for activity, and use callback functions to handle the messages that come in.

Channels

Here is a list of the channels used.

Channel	Published by	Description
/service/etws/join	client	Used for subscribing to channels, i.e., the order update channel and the two error channels
/etws/error	service	Used for higher-level error messages
/service/etws/error	service	Used for errors specific to the subscription
/service/etws/accountlisten	client	Used for specifying which accounts to listen to for order activity
/service/etws/accountunlisten	client	Used for specifying accounts to stop listening to
/service/etws/orderupdate	service	Used by service to transmit order status updates

Connecting requires that you publish to `/service/etws/join` with one of these two options:

Action	Description
join	<code>cometd.publish("/service/etws/join", { type: 'join' });</code>
reconnect	<code>cometd.publish("/service/etws/join", { type: 'reconnect' });</code>

Sample code

The following JavaScript examples demonstrate the basics of:

- connecting to the service
- subscribing to error messages and order status updates
- listening to specific accounts for activity
- unlistening to specific accounts

These samples require the CometD JavaScript library to work.

Connecting to the Service

This code establishes and re-establishes the connection to the service.

```

function _initialize()
{
    _cometd = $.cometd; // Use the default Comet object
    _cometUrl = "https://etwspush.etrade.com/apistream/cometd/oauth/";
    _metaSubscriptions = [];
    $.each(_metaSubscriptions, function(index, subscription) {
        _cometd.removeListener(subscription)
    });
    _metaSubscriptions.push(_cometd.addListener('/meta/handshake', this,
    _metaHandshake));
    _metaSubscriptions.push(_cometd.addListener('/meta/connect', this, _metaConnect));
    var oauthHeader = {Authorization: header};
    _cometd.init({url: _cometUrl, requestHeaders: oauthHeader});
}

function _metaHandshake(message)
{
    _handshook = message.successful;
    _connected = false;
}

function _metaConnect(message)
{
    var wasConnected = _connected;
    _connected = message.successful;
    if (wasConnected)
    {
        if (_connected)
        {
            // Normal operation, a long poll that reconnects
        }
        else
        {
            // Disconnected
        }
    }
    else
    {
        if (_connected)
        {
            // Connected
            baseSubscribe();
            if(_firstConnection){
                _cometd.publish("/service/etws/join", { type:'join'});
            }else{
                _cometd.publish("/service/etws/join", { type:'re-connect'});
            }
            _firstConnection = false;
        }
        else
        {
            // Could not connect
        }
    }
}

```

```
}
```

Handling Errors and Updates

This sample shows how callback functions are used to process messages from the `error` and `orderupdate` channels, described in the table of channels above.

```
function handleError(message)
{
    //Handle error
}

function handleUpdate(message)
{
    //Handle order status update message
}
```

Subscribing to Notification Channels

This sample code subscribes to the channels for system errors, service errors, and order status updates, using the `subscribe()` method from the Comet JavaScript library several times in a batch.

```
function baseSubscribe(){
    _cometd.startBatch();
    channel = _cometd.subscribe("/etws/error", this, handleError);
    channel = _cometd.subscribe("/service/etws/error", this, handleError);
    channel = _cometd.subscribe("/service/etws/orderupdate", this, handleUpdate);
    _cometd.endBatch();
}
```

Listening to Specific Accounts

Subscribing to the order-update channel is not sufficient for receiving order status updates. You must also specify which accounts to listen to. This is done by publishing to `accountlisten`. Specify a single account or a comma-separated list of accounts under the name "accounts", as shown in these examples.

Examples

```
cometd.publish("/service/etws/accountlisten", {accounts:83405188});
cometd.publish("/service/etws/accountlisten", {accounts:83405188,83405557,83412346});
```


Unlistening to Specific Accounts

To stop listening to order status updates for one or more accounts, publish to `accountunlisten`. Specify a single account or a comma-separated list of accounts under the name "accounts", as shown in these examples.

Examples

```
cometd.publish("/service/etws/accountunlisten", {accounts:83405188});  
cometd.publish("/service/etws/accountunlisten", {accounts:83405188,83405557,83412346})
```

Sample Response

Here is a sample order update message:

```
{  
  "class":"com.etrade.lmq.orderstatus.common.dto.OrderStatusResponse",  
  "quantityFilled":0,  
  "accountNumber":"83531504",  
  "price":0,  
  "symbol":"AAPL",  
  "legStatus":"OPEN",  
  "orderNumber":130,  
  "priceType":"TRIGGER_UNSPECIFIED",  
  "orderAction":"BUY",  
  "quantity":5,  
  "eventType":"OPEN",  
  "orderStatus":"OPEN"  
}
```

REST API Reference

This section contains detailed reference material on each of the individual REST APIs, organized into the following sub-sections:

- [Authorization](#)
- [Accounts](#)
- [Market](#)
- [Orders](#)
- [Rate Limits](#)
- [Notifications](#)
- [Error Handling](#)

Authorization

Get Request Token

Returns a temporary request token, initiating the OAuth process.

Description

This API returns a temporary request token that begins the OAuth process. The request token must accompany the user to the authorization page, where the user will grant your application limited access to the account. The token expires after five minutes.

URL

https://etws.etrade.com/oauth/request_token

HTTP Method: GET

Request Parameters

Property	Type	Description
oauth_consumer_key	string	The value used by the consumer to identify itself to the service provider.
oauth_timestamp	integer	The date and time of the request, in epoch time. Must be accurate within five minutes.
oauth_nonce	string	A nonce, as described in the authorization guide - roughly, an arbitrary or random value that cannot be used again with the same timestamp.
oauth_signature_method	string	The signature method used by the consumer to sign the request. The only supported value is "HMAC-SHA1".
oauth_signature	string	Signature generated with the shared secret and token secret using the specified <code>oauth_signature_method</code> , as described in OAuth documentation.
oauth_callback	string	Callback information, as described elsewhere. Must always be set to "oob", whether using a callback or not.

Response Properties

Property	Type	Description
oauth_token	string	The consumer's request token
oauth_token_secret	string	The token secret
oauth_callback_confirmed	boolean	Returns true if a callback URL is configured for the current consumer key, otherwise false. Callbacks are described under the Authorize Application API.

Sample Request

URL

```
GET https://etws.etrade.com/oauth/request_token
```

HTTP header

```
Authorization: OAuth  
realm="",oauth_callback="oob",oauth_signature="FjoSQaFDKEDK1FJazlY3xArNflk%3D",  
oauth_nonce="LTg2ODUzOTQ5MTEzMTY3MzQwMzE%3D",oauth_signature_method="HMAC-  
SHA1",oauth_consumer_key="282683cc9e4b8fc81dea6bc687d46758",oauth_timestamp="127325442  
5"
```

Sample Response

The response data is contained in the body of the response, URL-encoded. (This encoding type is identified in the response header, just as with XML and JSON responses.)

```
oauth_token=%2FiQRgQCRGPo7Xdk6G8QDSEzX0Jsy6sKNcULcDavAGgU%3D&oauth_token_secret=%2FrC9  
scEpzcwSEMy4vE7nodSzPLqfRINnTNY4voczyFM%3D&oauth_callback_confirmed=true
```

Notes

- The request token is only valid for 5 minutes.

Authorize Application

Allows the user to authorize the consumer application.

Description

Once your application has the request token, it should redirect the user to an E*TRADE authorization page, as shown in the URL below. Include the request token and your consumer key as parameters in the URL as shown. The page at this URL asks the user to authorize your application. Upon approval of the authorization request, E*TRADE generates a verification code.

By default, E*TRADE then displays this verification code in an "Authorization Complete" page, allowing the user to manually copy the code and paste it into your application. However, the recommended alternative is for E*TRADE to pass the verification code directly into your application via a pre-configured callback URL.

Using a callback requires that the callback URL be associated with your consumer key in the E*TRADE system. To request this, follow the instructions in our Authorization guide chapter (separate from this API reference). Your callback URL can be just a simple address, or can also include query parameters. Once the callback is configured, users who approve the authorization request are automatically redirected to the specified URL, with the verification code appended as a query parameter. Examples are shown in the Sample Response below.

URL

`https://us.etrade.com/e/etws/authorize?key={oauth_consumer_key}&token={oauth_token}`

HTTP Method: GET

Request Parameters

Property	Type	Required?	Description
oauth_consumer_key	string	required	The value used by the consumer to identify itself to the service provider.
oauth_token	string	required	The consumer's request token.

Response Properties

Property	Type	Description
oauth_verifier	query parameter	verification code

Sample Request

```
https://us.etrade.com/e/etws/authorize?key=282683cc9e4b8fc81dea6bc687d46758&token=%2FiQRgQCRGPo7Xdk6G8QDSEzX0Jsy6sKNcULcDavAGgU%3D
```

Sample Response

As described above, the `authorize` call is not a REST API in the usual sense, and does not return a "response" in the usual way. If the user authorizes your application on the E*TRADE authorization site, the result is either the display of a verification code at that site or, if a callback is used, a redirect to your callback URL.

In the callback scenario, the verification code is appended to your callback URL as an `oauth_verifier` parameter. Here are two examples:

```
https://myapplicationsite.com/mytradingapp?oauth_verifier=WXYZ89  
https://myapplicationsite.com?myapp=trading&oauth_verifier=WXYZ89
```

Notes

- If using the default approach - letting the user copy and paste the verification code - we recommend opening a separate browser window for the authorization, leaving your application open in the original window. Once the user has authorized your application and copied the verification code, the user can simply close the authorization window and return to your application in the original window.
- If using the callback approach, we recommend redirecting to the authorization page in the current window. Once the user has authorized your application, E*TRADE redirects the user to your callback page; the verification code is included as a URL parameter (as in the sample response above).

Get Access Token

Returns an OAuth access token.

Description

This method returns an access token, which confirms that the user has authorized the application to access user data.

All calls to the E*TRADE API (e.g., accountlist, placeequityorder, etc.) must include this access token along with the consumer key, timestamp, nonce, signature method, and signature. This can be done in the query string, but is typically done in the HTTP header.

Token lifespan

By default, the access token expires at the end of the current calendar day, US Eastern time. Once the token has expired, no requests will be processed for that token until the OAuth process is repeated - i.e., the user must log in again and the application must secure a new access token.

During the current day, if the application does not make any requests for two hours, the access token is inactivated. In this inactive state, the access token is not valid for authorizing requests. It must be reactivated using the Renew Access Token API.

URL

https://etws.etrade.com/oauth/access_token

HTTP Method: GET

Request Parameters

Property	Type	Required?	Description
oauth_consumer_key	string	required	The value used by the consumer to identify itself to the service provider.
oauth_timestamp	integer	required	The date and time of the request, in epoch time. Must be accurate to within five minutes.
oauth_nonce	string	required	A nonce, as described in the authorization guide - roughly, an arbitrary or random value that cannot be used again with the same timestamp.
oauth_signature_method	string	required	The signature method used by the consumer to sign the request. The only supported value is "HMAC-SHA1".

oauth_signature	string	required	Signature generated with the shared secret and token secret using the specified <code>oauth_signature_method</code> , as described in OAuth documentation.
oauth_token	string	required	The consumer's request token to be exchanged for an access token.
oauth_verifier	string	required	The code received by the user to authenticate with the third-party application.

Response Properties

Property	Type	Description
oauth_token	string	The consumer's access token
oauth_token_secret	integer	The token secret

Sample Request

URL

```
GET https://etws.etrade.com/oauth/access_token
```

HTTP header

```
Authorization: OAuth
realm="",oauth_signature="FjoSQaFDKEDK1FJazlY3xArNflk%3D",oauth_nonce="LTg2ODUzOTQ5MTE
zMTY3MzQwMzE%3D",oauth_signature_method="HMAC-SHA1",oauth_consumer_key=
"282683cc9e4b8fc81dea6bc687d46758",oauth_timestamp="1273254425",oauth_verifier="Y27X25
F",oauth_token=%2FiQRgQCRGPo7Xdk6G8QDSEzX0Jsy6sKNcULcDavAGgU%3D
```

Sample Response

```
oauth_token=%3TiQRgQCRGPo7Xdk6G8QDSEzX0Jsy6sKNcULcDavAGgU%3D&oauth_token_secret=%7RrC9
scEpzcwSEMy4vE7nodSzPLqfRINnTNY4voczyFM%3D
```

Notes

- The production access token expires by default at midnight US Eastern time.
- Technically, the access token and related parameters can be passed with HTTP requests as part of the URL, but we recommend this information be passed in the header of the request instead.

Renew Access Token

Renews the OAuth access token after two hours or more of inactivity.

Description

If the application does not make any requests for two hours, the access token is inactivated. In this inactive state, the access token is not valid for authorizing requests. It must be reactivated using the Renew Access Token API.

By default the access token expires at midnight US Eastern time. Once the token has expired, no further requests will be processed until the user logs in again and the application secures a new access token.

URL

https://etws.etrade.com/oauth/renew_access_token

HTTP Method: GET

Request Parameters

Property	Type	Required?	Description
oauth_consumer_key	string	required	The value used by the consumer to identify itself to the service provider.
oauth_timestamp	integer	required	The date and time of the request, in epoch time. Must be accurate within five minutes.
oauth_nonce	string	required	A nonce, as described in the authorization guide - roughly, an arbitrary or random value that cannot be used again with the same timestamp.
oauth_signature_method	string	required	The signature method used by the consumer to sign the request. The only supported value is "HMAC-SHA1".
oauth_signature	string	required	Signature generated with the shared secret and token secret using the specified <code>oauth signature method</code> , as described in OAuth documentation.
oauth_token	string	required	The consumer's request token to be exchanged for an access token.

Response Properties

Property	Type	Description
oauth_token	string	The consumer's access token.
oauth_token_secret	integer	The token secret.

Sample Request

URL

```
GET https://etws.etrade.com/oauth/renew_access_token
```

HTTP header

```
Authorization: OAuth
realm="",oauth_signature="FjoSQaFDKEDK1FJazlY3xArNflk%3D",oauth_nonce="LTg2ODUzOTQ5MTE
zMTY3MzQwMzE%3D",oauth_signature_method="HMAC-SHA1",oauth_consumer_key=
"282683cc9e4b8fc81dea6bc687d46758",oauth_timestamp="1273254425",oauth_token=%2FiQRgQCR
GPo7Xdk6G8QDSEzX0Jsy6sKNcULcDavAGgU%3D
```

Sample Response

```
oauth_token=%3TiQRgQCRGPo7Xdk6G8QDSEzX0Jsy6sKNcULcDavAGgU%3D&oauth_token_secret=%7RrC9
scEpzcwSEMy4vE7nodSzPLqfRINnTNY4voczyFM%3D
```

Revoke Access Token

Revokes an OAuth access token.

Description

This method revokes an access token that was granted for the consumer key. Once the token is revoked, it no longer grants access to E*TRADE data. We strongly recommend revoking the access token once your application no longer needs access to the user's E*TRADE account. In the event of a security compromise, a revoked token is useless to a malicious entity.

URL

https://etws.etrade.com/oauth/revoke_access_token

HTTP Method: GET

Request Parameters

Property	Type	Required?	Description
oauth_consumer_key	string	required	The value used by the consumer to identify itself to the service provider.
oauth_timestamp	integer	required	The date and time of the request, in epoch time. Must be accurate within five minutes.
oauth_nonce	string	required	A nonce, as described in the authorization guide - roughly, an arbitrary or random value that cannot be used again with the same timestamp.
oauth_signature_method	string	required	The signature method used by the consumer to sign the request. The only supported value is "HMAC-SHA1".
oauth_signature	string	required	Signature generated with the shared secret and token secret using the specified <code>oauth_signature_method</code> , as described in OAuth documentation.
oauth_token	string	required	The consumer's access token to be revoked.

Response Properties

Property	Type	Description
oauth_token	string	The consumer's access token
oauth_token_secret	integer	The token secret

Sample Request

URL

```
GET https://etws.etrade.com/oauth/revoke_access_token
```

HTTP header

```
Authorization: OAuth  
realm="",oauth_signature="FjoSQaFDKEDK1FJazlY3xArNflk%3D",oauth_nonce="LTg2ODUzOTQ5MTE  
zMTY3MzQwMzE%3D",oauth_signature_method="HMAC-SHA1",oauth_consumer_key=  
"282683cc9e4b8fc81dea6bc687d46758",oauth_timestamp="1273254425",oauth_token=%2FiQRgQCR  
GPo7Xdk6G8QDSEzX0Jsy6sKNcULcDavAGgU%3D
```

Sample Response

```
oauth_token=%3TiQRgQCRGPo7Xdk6G8QDSEzX0Jsy6sKNcULcDavAGgU%3D&oauth_token_secret=%7RrC9  
scEpzcwSEMy4vE7nodSzPLqfRINnTNY4voczyFM%3D
```

Accounts

List Accounts

Retrieves a list of the current user's E*TRADE brokerage accounts.

Description

This API returns a list of E*TRADE accounts for the current user. Only brokerage accounts are returned - that is, accounts that can be used for trading, as opposed to credit cards, etc.

URL

<https://etws.etrade.com/accounts/rest/accountlist>

HTTP Method: GET

Request Parameters

No parameters.

Response Properties

Property	Type	Description
accountDesc	string	A text description of the account
accountId	integer	Numeric account ID
marginLevel	string	The account's margin level. Possible values are: CASH, MARGIN.
netAccountValue	double	The total value of the account. This includes cash plus stocks, options, mutual funds, and bonds.
registrationType	string	The type of account. Possible values are: INDIVIDUAL, JOINT, CORPORATE, IRA, ESTATE, TRUST.

Sample Request

```
GET https://etws.etrade.com/accounts/rest/accountlist
```

Sample response - XML

```
<AccountListResponse>
  <Account>
    <accountDesc>MyAccount-1</accountDesc>
    <accountId>83405188</accountId>
    <marginLevel>MARGIN</marginLevel>
    <netAccountValue>9999871.82</netAccountValue>
    <registrationType>INDIVIDUAL</registrationType>
  </Account>
</Account>
```

```

    <accountDesc>MyAccount-3</accountDesc>
    <accountId>83405553</accountId>
    <marginLevel>CASH</marginLevel>
    <netAccountValue>100105468.99</netAccountValue>
    <registrationType>INDIVIDUAL</registrationType>
  </Account>
  <Account>
    <accountDesc>SIMPLE IRA</accountDesc>
    <accountId>83405188</accountId>
    <marginLevel>CASH</marginLevel>
    <netAccountValue>99794.13</netAccountValue>
    <registrationType>IRA</registrationType>
  </Account>
</AccountListResponse>

```

Sample response - JSON

```

{
  "AccountListResponse": {
    "Account": [
      {
        "accountDesc": "MyAccount-1",
        "accountId": "83405188",
        "marginLevel": "MARGIN",
        "netAccountValue": "9999871.82",
        "registrationType": "INDIVIDUAL"
      },
      {
        "accountDesc": "MyAccount-3",
        "accountId": "83405553",
        "marginLevel": "CASH",
        "netAccountValue": "100105468.99",
        "registrationType": "INDIVIDUAL"
      },
      {
        "accountDesc": "SIMPLE IRA",
        "accountId": "83405188",
        "marginLevel": "CASH",
        "netAccountValue": "99794.13",
        "registrationType": "IRA"
      }
    ]
  }
}

```

Sample use cases

Some possible use cases and workflows are described below.

Purpose	Workflow	Related APIs
---------	----------	--------------

List accounts, account detail	Display the type, description, and net value for each of the user's accounts as well as a detail display for each account.	List Accounts, Get Account Balance, Get Account Positions
Set the active account for a trade	When setting up an order, allow the user to select an account; this is critical when a user has multiple accounts and wants to place an order in a specific account (e.g., IRA rather than day-trading account).	List Accounts, Get Account Balance
Retrieve account summary info for order lists, order previews, and order validation	When listing orders, previewing an order, or validating an order before placing it, use List Accounts to look up the account type, description, and balance(s).	Get Account Balance, List Orders, Preview Equity Order, Preview Option Order, Preview Equity Order Change, Preview Option Order Change

Related APIs

Get Account Balance, Get Account Positions

Get Account Balance

Retrieves the current account balance and related details for a specified account.

Description

This API returns detailed balance information for a specified account for the current user. The information returned includes account type, option level, and details on up to four balances - account balance, margin account balance, day trade balance, and cash account balance.

URL

`https://etws.etrade.com/accounts/rest/accountbalance/{accountId}`

HTTP Method: GET

Request Parameters

Property	Type	Required?	Description
accountId	path	required	Numeric account ID

Sample Request

```
GET https://etws.etrade.com/accounts/rest/accountbalance/83405188
```

Response Properties

Property	Type	Description
accountId	integer	Numeric account ID
accountType	string	The account's margin level. Possible values are: CASH, MARGIN.
optionLevel	string	The option level of the account, i.e., what option-related actions are allowed. Possible values are: <ul style="list-style-type: none">• LEVEL_1 - Write covered calls.• LEVEL_2 - Write covered calls, purchases.• LEVEL_3 - Write covered calls, purchases, spreads, uncovered puts. (Margin or IRA account required for spreads; margin account required for uncovered puts.)• LEVEL_4 - Write covered calls, purchases, spreads, uncovered puts, uncovered equity/index calls, and uncovered index puts. (Margin account required.)
accountBalance	complex	Container for basic brokerage account balances
cashAvailableForWithdrawal	double	The amount of cash that is available for withdrawal

cashCall	double	If a cash call has been issued, this value is the amount of cash or equivalent securities needed to meet the cash call. For a definition of cash call, refer to the E*TRADE online glossary .
fundsWithheldFromPurchasePower	double	The amount of any funds withheld from the purchasing power, if any
fundsWithheldFromWithdrawal	double	The total of any funds that are not allowed to be withdrawn
netAccountValue	double	The account value minus any margin debt
netCash	double	The net cash amount held in the account. A negative balance reflects margin debt.
sweepDepositAmount	double	The amount held in the sweep deposit account
totalLongValue	double	The total value of securities that are held long in the account
totalSecuritiesMktValue	double	The total value of the securities held in the account
totalCash	double	The total cash amount held in the account. A negative balance reflects margin debt.
marginAccountBalance	complex	Container for margin-related account balances. Only appears for margin-level accounts.
fedCall	double	If funds are required to meet a Regulation T "fed call", this value is the amount the investor must deposit for buying on margin or selling short. For a definition of fed call, refer to the E*TRADE online glossary .
marginBalance	double	The net open balance in the user's margin account
marginBalanceWithdrawal	double	The amount of the margin balance that is available for withdrawal
marginEquity	double	The value of the account's margin positions minus any margin debt
marginEquityPct	double	The account's equity divided by the market value of its marginable securities
marginableSecurities	double	The value of marginable securities held in the account
maxAvailableForWithdrawal	double	The total amount of cash plus margin equity that is available for withdrawal from this account
minEquityCall	double	If the account's margin equity has fallen below E*TRADE's minimum account equity requirement, this value is the amount of additional equity required. For a definition of minimum equity call, refer to the E*TRADE online glossary .
nonMarginableSecuritiesAnd Options	double	The value of securities and options that are not marginable
totalShortValue	double	The total value of securities that are held short in the account

shortReserve	double	The amount reserved to cover the short positions in the account
dtBalance	complex	Container for account balances related to day-trading. Only appears for day-trading accounts.
dtCashBalance	double	The cash available for investment in a day-trading account
dtMarginBalance	double	The margin available for investment in a day-trading account
dtMarginableSecurities	double	The value of marginable securities held in a day-trading account
dtNonMarginableSecuritiesAndOptions	double	The value of non-marginable securities and options held in a day-trading account
dtStatus	string	The current status of a day-trading account. Possible values are: <ul style="list-style-type: none"> • NOT_QUALIFIED • QUALIFIED_4X • MARGIN_CALL_2XD • MARGIN_CALL_1XD • MIN_EQUITY_CALL_1XK • QUALIFIED_RESTRICTION • CASH_ONLY
cashAccountBalance	complex	Container for account cash balances
cashAvailableForInvestment	double	The total amount of cash available for investment in the account
cashBalance	double	The amount of cash in the account
settledCashForInvestment	double	The amount of settled cash in the account
unSettledCashForInvestment	double	The amount of unsettled cash in the account

Sample response - XML

```
<AccountBalanceResponse>
  <accountId>83405188</accountId>
  <accountType>MARGIN</accountType>
  <optionLevel>LEVEL_4</optionLevel>
  <accountBalance>
    <cashAvailableForWithdrawal>0.00</cashAvailableForWithdrawal>
    <cashCall>0.00</cashCall>
    <fundsWithheldFromPurchasePower>0.00</fundsWithheldFromPurchasePower>
    <fundsWithheldFromWithdrawal>0.00</fundsWithheldFromWithdrawal>
    <netAccountValue>-286.55</netAccountValue>
    <netCash>-286.55</netCash>
    <sweepDepositAmount>0.00</sweepDepositAmount>
    <totalLongValue>0.00</totalLongValue>
    <totalSecuritiesMktValue>0.00</totalSecuritiesMktValue>
    <totalCash>-286.55</totalCash>
  </accountBalance>
  <marginAccountBalance>
    <fedCall>0.00</fedCall>
    <marginBalance>-286.55</marginBalance>
  </marginAccountBalance>
</AccountBalanceResponse>
```

```

    <marginBalanceWithdrawal>-286.55</marginBalanceWithdrawal>
    <marginEquity>-286.55</marginEquity>
    <marginEquityPct>0.00</marginEquityPct>
    <marginableSecurities>-1146.20</marginableSecurities>
    <maxAvailableForWithdrawal>-286.55</maxAvailableForWithdrawal>
    <minEquityCall>2286.55</minEquityCall>
    <nonMarginableSecuritiesAndOptions>-286.55</nonMarginableSecuritiesAndOptions>
    <totalShortValue>0.00</totalShortValue>
    <shortReserve>0.00</shortReserve>
  </marginAccountBalance>
</AccountBalanceResponse>

```

Sample response - JSON

```

{
  "AccountBalanceResponse": {
    "accountId": "83405188",
    "accountType": "MARGIN",
    "optionLevel": "LEVEL_4",
    "accountBalance": {
      "cashAvailableForWithdrawal": "0.00",
      "cashCall": "0.00",
      "fundsWithheldFromPurchasePower": "0.00",
      "fundsWithheldFromWithdrawal": "0.00",
      "netAccountValue": "-286.55",
      "netCash": "-286.55",
      "sweepDepositAmount": "0.00",
      "totalLongValue": "0.00",
      "totalSecuritiesMktValue": "0.00",
      "totalCash": "-286.55"
    },
    "marginAccountBalance": {
      "fedCall": "0.00",
      "marginBalance": "-286.55",
      "marginBalanceWithdrawal": "-286.55",
      "marginEquity": "-286.55",
      "marginEquityPct": "0.00",
      "marginableSecurities": "-1146.20",
      "maxAvailableForWithdrawal": "-286.55",
      "minEquityCall": "2286.55",
      "nonMarginableSecuritiesAndOptions": "-286.55",
      "totalShortValue": "0.00",
      "shortReserve": "0.00"
    }
  }
}

```

Sample use cases

Some possible use cases and workflows are described below.

Purpose	Workflow	Related APIs
---------	----------	--------------

list of accounts, account detail	Display the type, description, and net value for each of the user's accounts as well as a detail display for each account, including balances and positions.	List Accounts, Get Account Balance, Get Account Positions
balance summary	When previewing or validating an order before placing it, use Get Account Balance to check the account balances for sufficient funds.	Get Account Balance, List Orders, Preview Equity Order, Preview Option Order, Preview Equity Order Change, Preview Option Order Change

Related APIs

List Accounts, Get Account Positions, Get Transaction History, Get Transaction Details

Sandbox Samples

The only parameter for this API is the account ID, which is required. The following shows a typical request and response in the sandbox environment.

Request

```
GET https://etwssandbox.etrade.com/accounts/sandbox/rest/accountbalance/83405188.json
```

Response

```
{
  "AccountBalanceResponse": {
    "accountId": "83405188",
    "accountType": "MARGIN",
    "optionLevel": "LEVEL_4",
    "accountBalance": {
      "cashAvailableForWithdrawal": "9986621.36",
      "fundsWithheldFromWithdrawal": "0.00",
      "netAccountValue": "10086354.52",
      "netCash": "9997973.185855",
      "sweepDepositAmount": "0.00",
      "totalLongValue": "88255.00",
      "totalSecuritiesMktValue": "88255.00"
    },
    "marginAccountBalance": {
      "marginBalance": "9997973.19",
      "marginBalanceWithdrawal": "9986621.36",
      "marginEquity": "9989687.81",
      "marginableSecurities": "19978072.55",
      "maxAvailableForWithdrawal": "9986621.36",
      "nonMarginableSecuritiesAndOptions": "9986621.36",
      "shortReserve": "126.33"
    }
  }
}
```

Get Account Positions

Retrieves the positions held in the specified account.

Description

This API returns information on positions held in an account. The response includes the total count of positions and information about each one - market symbol, type, strike price, expiration date.

Since an account may have a large number of positions, the API allows you to page through them in small groups by specifying how many to return at a time (the `count`) and the starting point for each group (the `marker`).

URL

<https://etws.etrade.com/accounts/rest/accountpositions/{accountId}>

HTTP Method: GET

Request Parameters

The only required parameter is the `accountId`. If the `typeCode` is `OPTN`, you must specify some additional parameters: the type code, `CALL` or `PUT`, the strike price, and the expiration date.

Property	Type	Required?	Description
<code>accountId</code>	path	required	Numeric account ID
<code>count</code>	integer	optional	The number of positions to return in the response. If not specified, defaults to 25. Used for paging as described in the Notes below.
<code>marker</code>	string	optional	Specifies the desired starting point of the set of items to return. Used for paging as described in the Notes below.
<code>typeCode</code>	string	optional	The type of security. Possible values are: <code>EQ</code> (equity), <code>OPTN</code> (option), <code>INDX</code> (index), <code>MF</code> (mutual fund), <code>FI</code> (fixed income). If set to <code>OPTN</code> , must specify <code>callPut</code> , <code>strikePrice</code> , <code>expYear</code> , <code>expMonth</code> , and <code>expDay</code> .
<code>symbol</code>	string	conditional	The market symbol. Required if <code>typeCode</code> is <code>OPTN</code> .
<code>callPut</code>	enum	conditional	Specifies which type of option(s) to return. Possible values are: <code>CALL</code> or <code>PUT</code> . Required if <code>typeCode</code> is <code>OPTN</code> .
<code>strikePrice</code>	double	conditional	Specifies the <code>strikePrice</code> of the desired option. Required if <code>typeCode</code> is <code>OPTN</code> .
<code>expYear</code>	string	conditional	The year the option will expire, as specified in the option contract. Required if <code>typeCode</code> is <code>OPTN</code> .
<code>expMonth</code>	integer	conditional	The year the option will expire, as specified in the option contract. Required if <code>typeCode</code> is <code>OPTN</code> .

expDay	integer	conditional	The year the option will expire, as specified in the option contract. Required if typeCode is OPTN.
--------	---------	-------------	---

Sample Request 1 - Equities

```
GET https://etws.etrade.com/account/rest/accountpositions/83405188
```

Sample Request 2 - Options

```
GET
https://etws.etrade.com/account/rest/accountpositions/83405188?typeCode=OPTN&symbol=GO
OG&callPut=CALL&strikePrice=100&expYear=2011&expMonth=1&expDay=1
```

Response Properties

Property	Type	Description
accountId	string	Numeric account ID
count	integer	The number of positions contained in the response. If a count is not specified in the request, the defaults to a maximum of 25.
marker	string	Specifies the starting point for the next set of positions, if any. This property is empty if there are no more positions in the account after these.
AccountPositions	complex	Container for one or more account position elements
AccountPosition	complex	Container for information on a position. Only appears if the account has at least one position.
costBasis	double	The amount the user paid for this position
description	string	Text description of the position (e.g., company name)
longOrShort	string	Whether the position is long or short. Possible values are: LONG, SHORT.
marginable	boolean	Boolean that specifies whether the position counts toward your margin (true) or does not count toward your margin (false)
qty	double	The number of shares held in this position. May include fractional shares.
currentPrice	double	Current market price (last sale price)
closePrice	double	Previous day's closing price
marketValue	double	The value of a position, based on the quantity and previous day's closing price
quoteType	string	Expiry type, if the product is an option. Possible values are: QUARTERLY, MONTHLY, or WEEKLY.
adjustedOption	boolean	If TRUE, the option has been adjusted. Default is FALSE.
deliverableStr	string	Display-formatted product identifier. For equities, this is a symbol; for options, this is a string containing symbol, "call" or "put", strike price, and expiration date.
productId	complex	Container for product information for this position

symbol	string	The market trading symbol for the option
typeCode	string	The type of security. Possible values are: EQ (equity), OPTN (option), INDX (index), MF (mutual fund), FI (fixed income).
callPut	string	Specifies whether options are being bought (call) or sold (put), as specified in the option order. Possible values are: CALL, PUT. Appears for options only.
strikePrice	double	The strike price for the option. Appears for options only.
expYear	integer	The year the option will expire. Appears for options only.
expMonth	integer	The month (1-12) the option will expire. Appears for options only.
expDay	integer	The day (1-31) the option will expire. Appears for options only.

Sample Response 1 - XML

```

<AccountPositionsResponse>
  <accountId>30049872</accountId>
  <count>2</count>
  <marker></marker>
  <AccountPositions>
    <AccountPosition>
      <costBasis>63.10</costBasis>
      <description>DU PONT E I DE NEMOURS & CO COM</description>
      <longOrShort>LONG</longOrShort>
      <marginable>true</marginable>
      <productId>
        <symbol>DD</symbol>
        <typeCode>EQ</typeCode>
      </productId>
      <qty>10</qty>
      <currentPrice>49.76</currentPrice>
      <closePrice>49.34</closePrice>
      <marketValue>497.60</marketValue>
      <quoteType>2</quoteType>
      <adjustedOption>0</adjustedOption>
      <deliverableStr></deliverableStr>
    </AccountPosition>
    <AccountPosition>
      <costBasis>192.00</costBasis>
      <description>DELL INC COM</description>
      <longOrShort>LONG</longOrShort>
      <marginable>true</marginable>
      <productId>
        <symbol>DELL</symbol>
        <typeCode>EQ</typeCode>
      </productId>
      <qty>20</qty>
      <currentPrice>9.59</currentPrice>
      <closePrice>9.55</closePrice>
      <marketValue>191.80</marketValue>
      <quoteType>2</quoteType>
    </AccountPosition>
  </AccountPositions>
</AccountPositionsResponse>

```

```

        <adjustedOption>0</adjustedOption>
        <deliverableStr></deliverableStr>
    </AccountPosition>
</AccountPositions>
</AccountPositionsResponse>

```

Sample Response 1 - JSON

```

{
  "AccountPositionsResponse": {
    "accountId": "83405188",
    "count": "2",
    "AccountPositions": {
      "AccountPosition": [
        {
          "costBasis": "1096.44",
          "description": "GOOGLE INC CL A",
          "longOrShort": "LONG",
          "marginable": "FALSE",
          "productId": {
            "symbol": "GOOG",
            "typeCode": "EQ"
          },
          "qty": "2",
          "marketValue": "1086.44"
        },
        {
          "costBasis": "54.30",
          "description": "HELMERICH AND PAYNE INC COM",
          "longOrShort": "LONG",
          "marginable": "TRUE",
          "productId": {
            "symbol": "HP",
            "typeCode": "EQ"
          },
          "qty": "1",
          "marketValue": "44.30"
        }
      ]
    }
  }
}

```

Sample Response 2 - XML

```

<AccountPositionsResponse>
  <accountId>83405188</accountId>
  <count>1</count>
  <marker></marker>
  <AccountPositions>
    <AccountPosition>
      <costBasis>31.40</costBasis>
      <description>GOOG Feb 20 '10 $500 Call</description>

```



```

<longOrShort>SHORT</longOrShort>
<productId>
  <symbol>GOP</symbol>
  <typeCode>OPTN</typeCode>
  <callPut>CALL</callPut>
  <strikePrice>500</strikePrice>
  <expYear>2010</expYear>
  <expMonth>2</expMonth>
  <expDay>20</expDay>
</productId>
<qty>-1</qty>
<currentPrice>-3.32</currentPrice>
<marketValue>-3320.00</marketValue>
</AccountPosition>
</AccountPositions>
</AccountPositionsResponse>

```

Sample Response 2 - JSON

```

{
  "AccountPositionsResponse": {
    "accountId": "83405188",
    "count": "1",
    "AccountPositions": {
      "AccountPosition": {
        "costBasis": "31.40",
        "description": "GOOG Feb 20 '10 $500 Call",
        "longOrShort": "SHORT",
        "productId": {
          "symbol": "GOP",
          "typeCode": "OPTN",
          "callPut": "CALL",
          "strikePrice": "500",
          "expYear": "2010",
          "expMonth": "2",
          "expDay": "20"
        },
        "qty": "-1",
        "currentPrice": "-3.32",
        "marketValue": "-3320.00"
      },
      "qty": "-1",
      "currentPrice": "-3.32",
      "marketValue": "-3320.00"
    }
  }
}

```

Notes

- To page through a large number of items, use the `count` property to specify how many items to return in a group (the default is 25), and the `marker` property to specify the starting point (the default is the newest). For instance, a request with no `count` and no `marker` retrieves the newest 25 items. Each response includes a marker that points to the beginning of the next group. To page through all the items, repeat the request with

the marker from each previous response until you receive a response with an empty marker, indicating that there are no more items.

- The API does not explicitly provide for bi-directional paging. Your application can support paging backward and forward either by saving and re-using markers within the series (that is, to re-issue the requests for earlier pages in the series), or saving and re-displaying the items that are returned.

Sample use cases

Some possible use cases and workflows are described below.

Purpose	Workflow	Related APIs
List accounts, display account details	Display the type, description, and net value for each of the user's accounts as well as a detail display for each account, including balances and positions.	List Accounts, Get Account Balance, Get Account Positions
List and display positions	Navigate from a detail display of account positions to an order list to possibly place orders and modify positions.	List Orders, Get Account Positions

Related APIs

List Accounts, Get Account Balance, Get Transaction History, Get Transaction Details

The only required parameter for this API is account number.

Sandbox Samples

Request

```
GET
https://etwssandbox.etrade.com/accounts/sandbox/rest/accountpositions/83405188?typeCode=EQ&symbol=IBM
```

Response

```
<AccountPositionsResponse>
  <accountId>83405188</accountId>
  <count>11</count>
  <marker></marker>
  <AccountPositions>
    <AccountPosition>
      <costBasis>1096.44</costBasis>
      <description>GOOGLE INC CL A</description>
      <longOrShort>LONG</longOrShort>
      <marginable>>false</marginable>
      <productId>
        <symbol>GOOG</symbol>
        <typeCode>EQ</typeCode>
      </productId>
      <qty>2</qty>
```

```

    <marketValue>1086.44</marketValue>
  </AccountPosition>
  <AccountPosition>
    <costBasis>54.30</costBasis>
    <description>HELMERICH & PAYNE INC COM</description>
    <longOrShort>LONG</longOrShort>
    <marginable>true</marginable>
    <productId>
      <symbol>HP</symbol>
      <typeCode>EQ</typeCode>
    </productId>
    <qty>1</qty>
    <marketValue>44.30</marketValue>
  </AccountPosition>
  <AccountPosition>
    <costBasis>54.30</costBasis>
    <description>HELMERICH & PAYNE INC COM</description>
    <longOrShort>LONG</longOrShort>
    <marginable>true</marginable>
    <productId>
      <symbol>HP</symbol>
      <typeCode>EQ</typeCode>
    </productId>
    <qty>1</qty>
    <marketValue>44.30</marketValue>
  </AccountPosition>
  <AccountPosition>
    <costBasis>9595.75</costBasis>
    <description>INTERNATIONAL BUSINESS MACHS COM</description>
    <longOrShort>LONG</longOrShort>
    <marginable>true</marginable>
    <productId>
      <symbol>IBM</symbol>
      <typeCode>EQ</typeCode>
    </productId>
    <qty>75</qty>
    <marketValue>9585.75</marketValue>
  </AccountPosition>
  <AccountPosition>
    <costBasis>127.81</costBasis>
    <description>INTERNATIONAL BUSINESS MACHS COM</description>
    <longOrShort>SHORT</longOrShort>
    <marginable>true</marginable>
    <productId>
      <symbol>IBM</symbol>
      <typeCode>EQ</typeCode>
    </productId>
    <qty>-1</qty>
    <marketValue>-127.81</marketValue>
  </AccountPosition>
  <AccountPosition>
    <costBasis>2094.00</costBasis>
    <description>INTEL CORP COM</description>
    <longOrShort>LONG</longOrShort>

```

```

    <marginable>true</marginable>
    <productId>
      <symbol>INTC</symbol>
      <typeCode>EQ</typeCode>
    </productId>
    <qty>100</qty>
    <marketValue>2084.00</marketValue>
  </AccountPosition>
  <AccountPosition>
    <costBasis>67.94</costBasis>
    <description>MICROSOFT CORP COM</description>
    <longOrShort>LONG</longOrShort>
    <marginable>true</marginable>
    <productId>
      <symbol>MSFT</symbol>
      <typeCode>EQ</typeCode>
    </productId>
    <qty>2</qty>
    <marketValue>57.94</marketValue>
  </AccountPosition>
  <AccountPosition>
    <costBasis>12.00</costBasis>
    <description>A Mar 20 '10 $35 Call</description>
    <longOrShort>LONG</longOrShort>
    <productId>
      <symbol>A</symbol>
      <typeCode>OPTN</typeCode>
      <callPut>CALL</callPut>
      <strikePrice>35</strikePrice>
      <expYear>2010</expYear>
      <expMonth>3</expMonth>
      <expDay>20</expDay>
    </productId>
    <qty>2</qty>
    <marketValue>200.00</marketValue>
  </AccountPosition>
  <AccountPosition>
    <costBasis>31.40</costBasis>
    <description>GOOG Feb 20 '10 $500 Call</description>
    <longOrShort>SHORT</longOrShort>
    <productId>
      <symbol>GOP</symbol>
      <typeCode>OPTN</typeCode>
      <callPut>CALL</callPut>
      <strikePrice>500</strikePrice>
      <expYear>2010</expYear>
      <expMonth>2</expMonth>
      <expDay>20</expDay>
    </productId>
    <qty>-1</qty>
    <marketValue>-3320.00</marketValue>
  </AccountPosition>
  <AccountPosition>
    <costBasis>14.70</costBasis>
    <description>HP Feb 20 '10 $37.5 Call</description>

```

```

    <longOrShort>LONG</longOrShort>
    <productId>
      <symbol>HP</symbol>
      <typeCode>OPTN</typeCode>
      <callPut>CALL</callPut>
      <strikePrice>37.50</strikePrice>
      <expYear>2010</expYear>
      <expMonth>2</expMonth>
      <expDay>20</expDay>
    </productId>
    <qty>1</qty>
    <marketValue>470.00</marketValue>
  </AccountPosition>
  <AccountPosition>
    <costBasis>0.10</costBasis>
    <description>YUM Apr 17 &apos;10 $45 Call</description>
    <longOrShort>SHORT</longOrShort>
    <productId>
      <symbol>UUG</symbol>
      <typeCode>OPTN</typeCode>
      <callPut>CALL</callPut>
      <strikePrice>45</strikePrice>
      <expYear>2010</expYear>
      <expMonth>4</expMonth>
      <expDay>17</expDay>
    </productId>
    <qty>-1</qty>
    <marketValue>-5.00</marketValue>
  </AccountPosition>
</AccountPositions>
</AccountPositionsResponse>

```

List Alerts

Lists alerts for the current user.

Description

When called with HTTP GET and no parameters, this resource returns a list of alerts related to trading and account activity. For each alert, it returns an alert ID, the date and time the alert was issued, the subject, and a flag that shows whether the alert has ever been read.

When called with different syntax, the `alerts` resource is also used to read or delete an individual alert.

Order status alerts

The items returned by this API include messages about order status (completed, cancelled, etc.). However, to receive order status updates in a timely way, it's better to use the streaming API, which pushes the updates to the application immediately rather than waiting to be polled. For details, see our documentation on the Streaming API.

URL

<https://etws.etrade.com/accounts/rest/alerts>

HTTP Method: GET

Request Parameters

No parameters.

Sample Request

```
GET https://etws.etrade.com/accounts/rest/alerts
```

Response Properties

Property	Type	Description
Alerts	complex	Container for one or more <code>Alert</code> elements
Alert	complex	Container for an alert. Only appears if there is at least one alert for the user.
dateTime	long	The date and time the alert was issued, in epoch time
alertId	integer	Numeric alert ID
subject	string	Subject of the alert
readFlag	string	Indicates whether the alert has been read by the user (i.e., previously retrieved in detailed format by calling this API with this <code>alertId</code> as a parameter). Possible values are: <code>READ</code> , <code>UNREAD</code> .
symbol	string	Market symbol for the instrument related to this alert, if any, such as <code>GOOG</code>

Sample response - XML

```
<AlertsResponse>
  <Alerts>
    <Alert>
      <dateTime>1266519024</dateTime>
      <alertId>1108</alertId>
      <subject>Sell Short 1 HP Executed @ $2</subject>
      <readFlag>UNREAD</readFlag>
      <symbol></symbol>
    </Alert>
    <Alert>
      <dateTime>1266518336</dateTime>
      <alertId>1107</alertId>
      <subject>Buy to cover 1 HP Executed @ $50</subject>
      <readFlag>UNREAD</readFlag>
      <symbol></symbol>
    </Alert>
  </Alerts>
</AlertsResponse>
```

Sample response - JSON

```
{
  "AlertsResponse": {
    "Alerts": {
      "Alert": [
        {
          "dateTime": "1266519024",
          "alertId": "1108",
          "subject": "Sell Short 1 HP Executed @ $2",
          "readFlag": "UNREAD"
        },
        {
          "dateTime": "1266518336",
          "alertId": "1107",
          "subject": "Buy to cover 1 HP Executed @ $50",
          "readFlag": "UNREAD"
        }
      ]
    }
  }
}
```

Notes

- For timely order status updates, we suggest using the Streaming API.

Sample use cases

Some possible use cases and workflows are described below.

Purpose	Workflow	Related APIs
List active alerts	Display a list of new or recent alerts when starting the application and by polling at regular intervals. Allow the user to read or delete an alert by selecting it from the list.	List Alerts, Read Alert, Delete Alert

Related APIs

Read Alert, Delete Alert, Streaming API

Sandbox Samples

Request

```
GET https://etwssandbox.etrade.com/accounts/sandbox/rest/alerts
```

Response

```
<AlertsResponse>
  <Alerts>
    <Alert>
      <dateTime>1266519024</dateTime>
      <alertId>1108</alertId>
      <subject>Sell Short 1 HP Executed @ $2</subject>
      <readFlag>UNREAD</readFlag>
      <symbol></symbol>
    </Alert>
    <Alert>
      <dateTime>1266518336</dateTime>
      <alertId>1107</alertId>
      <subject>Buy to cover 1 HP Executed @ $50</subject>
      <readFlag>UNREAD</readFlag>
      <symbol></symbol>
    </Alert>
    <Alert>
      <dateTime>1266518271</dateTime>
      <alertId>1106</alertId>
      <subject>Buy to cover 1 HP Executed @ $50</subject>
      <readFlag>UNREAD</readFlag>
      <symbol></symbol>
    </Alert>
    <Alert>
      <dateTime>1266517369</dateTime>
      <alertId>1105</alertId>
      <subject>Buy 1 ETFC Executed @ $1</subject>
      <readFlag>UNREAD</readFlag>
      <symbol></symbol>
    </Alert>
    <Alert>
      <dateTime>1266516266</dateTime>
      <alertId>1104</alertId>
      <subject>Buy 4 CSCO Cancelled</subject>
```



```

    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266515972</dateTime>
    <alertId>1103</alertId>
    <subject>Buy 2 MSFT Cancelled</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498346</dateTime>
    <alertId>1102</alertId>
    <subject>Buy 1 QQQQ Executed @ $100</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498323</dateTime>
    <alertId>1101</alertId>
    <subject>Buy 1 MSFT Executed @ $0.1</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498302</dateTime>
    <alertId>1099</alertId>
    <subject>Buy 1 QQQQ Executed @ $100</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498295</dateTime>
    <alertId>1100</alertId>
    <subject>Buy 1 QQQQ Executed @ $88.3</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498293</dateTime>
    <alertId>1098</alertId>
    <subject>Buy 1 QQQQ Executed @ $0.1</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498287</dateTime>
    <alertId>1096</alertId>
    <subject>Sell 1 IBM Executed @ $127</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498281</dateTime>

```

```

    <alertId>1097</alertId>
    <subject>Sell 1 IBM Executed @ $1</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498276</dateTime>
    <alertId>1095</alertId>
    <subject>Buy 2 GOOG Executed @ $0.01</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498273</dateTime>
    <alertId>1094</alertId>
    <subject>Sell Short 1 IBM Executed @ $1</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498268</dateTime>
    <alertId>1093</alertId>
    <subject>Buy to cover 1 IBM Executed @ $1</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498242</dateTime>
    <alertId>1092</alertId>
    <subject>Buy 2 GOOG Triggered</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266498004</dateTime>
    <alertId>1091</alertId>
    <subject>Buy 1 MSFT Triggered</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266490639</dateTime>
    <alertId>1090</alertId>
    <subject>Buy 2 MSFT Cancelled</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>
    <dateTime>1266490294</dateTime>
    <alertId>1089</alertId>
    <subject>Buy 4 IBM Cancelled</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
</Alert>
<Alert>

```

```

    <dateTime>1266490249</dateTime>
    <alertId>1088</alertId>
    <subject>Buy 2 IBM Cancelled</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
  </Alert>
  <Alert>
    <dateTime>1266489623</dateTime>
    <alertId>1087</alertId>
    <subject>Buy 1220 IBM Cancelled</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
  </Alert>
  <Alert>
    <dateTime>1266488490</dateTime>
    <alertId>1086</alertId>
    <subject>Buy 1 IBM Executed @ $134</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
  </Alert>
  <Alert>
    <dateTime>1266486731</dateTime>
    <alertId>1085</alertId>
    <subject>Buy 1 HP Feb 50 Put Executed @ $5</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
  </Alert>
  <Alert>
    <dateTime>1266486647</dateTime>
    <alertId>1084</alertId>
    <subject>Sell 1 HP Feb 50 Put Executed @ $5</subject>
    <readFlag>UNREAD</readFlag>
    <symbol></symbol>
  </Alert>
</Alerts>
</AlertsResponse>

```

Read Alert

Retrieves a specified alert.

Description

When called using HTTP GET with an alert ID, this resource retrieves the details of the specified alert, including the date and time it was created, the subject and complete text of the alert, and the date when it was first read, if ever.

When called with different syntax, the `alerts` resource is also used to list the available alerts for the user or delete an individual alert.

URL

`https://etws.etrade.com/accounts/rest/alerts`

HTTP Method: GET

Request Parameters

Property	Type	Required?	Description
alertId	path	required	Numeric alert ID

Sample Request

```
GET https://etws.etrade.com/accounts/rest/alerts/1108
```

Response Properties

Property	Type	Description
createDate	long	Date and time when the alert was issued, in epoch time
alertId	integer	Numeric alert ID
msgText	string	Body of the alert
readDate	long	The date and time when the alert was first read (i.e., retrieved), in epoch time. When retrieved for the first time, this value is 0 (zero). After that, this value is permanently set to the time of the first retrieval.
subject	string	Subject of the alert

Sample response - XML

```
<AlertDetailsResponse>
  <createDate>1266608946</createDate>
  <alertId>1108</alertId>
  <msgText>Account: xxxx-5557 Your day Trailing Stop Buy to open order for 5 IBM Feb
110 Calls was cancelled See order # 183 for details.</msgText>
  <readDate>0</readDate>
```

```
<subject>Buy 5 IBM Feb 110 Calls Cancelled</subject>
</AlertDetailsResponse>
```

Sample response - JSON

```
{
  "AlertDetailsResponse": {
    "createDate": "1266608946",
    "alertId": "1108",
    "msgText": "Account: xxxx-5557 Your day Trailing Stop Buy to open order for 5 IBM Feb 110 Calls was cancelled See order # 183 for details.",
    "readDate": "0",
    "subject": "Buy 5 IBM Feb 110 Calls Cancelled"
  }
}
```

Notes

- The first retrieval of a particular alert will show 0 as the read date. On subsequent retrievals, the read date will always be the date & time of that first retrieval.

Sample use cases

A possible use case is described below.

Purpose	Workflow	Related APIs
Display alert details	Display the details of an alert - for instance, when the alert is selected from a list of available alerts.	List Alerts, Read Alert

Related APIs

List Alerts, Delete Alert, Streaming API

Sandbox Samples

Request

```
GET https://etwssandbox.etrade.com/accounts/sandbox/rest/alerts/1108
```

Response

```
<AlertDetailsResponse>
  <createDate>1266608946</createDate>
  <alertId>1108</alertId>
  <msgText>Account: xxxx-5557 Your day Trailing Stop Buy to open order for 5 IBM Feb 110 Calls was cancelled See order # 183 for details.</msgText>
  <readDate>0</readDate>
  <subject>Buy 5 IBM Feb 110 Calls Cancelled</subject>
</AlertDetailsResponse>
```


Delete Alert

Deletes a specified alert.

Description

When called using HTTP DELETE with an alert ID, this resource marks the specified alert as deleted. This effectively removes the alert from any further access by the user.

When called with different syntax, the `alerts` resource is also used to list the available alerts for the user or retrieve the details of an individual alert.

URL

<https://etws.etrade.com/accounts/rest/alerts>

HTTP Method: DELETE

Request Parameters

Property	Type	Required?	Description
alertId	path	required	Numeric alert ID

Sample Request

```
DELETE https://etws.etrade.com/accounts/rest/alerts/321
```

Response Properties

Property	Type	Description
result	string	Result of the operation. Possible values are: SUCCESS, FAILURE.

Sample response - XML

```
<deleteAlertResponse>
  <result>Success</result>
</deleteAlertResponse>
```

Sample response - JSON

```
{
  "json.deleteAlertResponse": {
    "result": "Success"
  }
}
```

Sample use cases

Some possible use cases and workflows are described below.

Purpose	Workflow	Related APIs
Delete an alert	Delete an alert - for example, when the alert is selected on a list and the user chooses "delete" from a menu or context menu.	List Alerts, Delete Alert
Delete multiple alerts	Use checkbox controls on an alert list to allow batch deletion of alerts.	Delete Alert

Related APIs

List Alerts, Read Alert

Sandbox Samples

Request

```
DELETE https://etws.etrade.com/accounts/rest/alerts/321.json
```

Response

```
{
  "json.deleteAlertResponse": {
    "result": "Success"
  }
}
```


Get Transaction History

Retrieves a list of transactions for a specified account.

Description

This API returns a list of transactions (such as deposits, withdrawals, and transfers) for a single, specified account. The request can specify start and end dates, going back as much as two years, and the maximum number of transactions to return. Results can be filtered by:

- a single transaction type (e.g., check, dividend, transfer)
- a group of related transaction types - deposits, withdrawals, or trades
- within a group (e.g., deposits), a list of transaction types (e.g., dividend & interest)
- within the trades group:
 - one or all asset types - equities, options, money market, mutual funds, or bonds
 - within a single asset type, a single ticker symbol

Since an account history may contain a large number of transactions, the API allows you to page through them in groups by specifying how many to return at a time (the `count`) and the starting point for each successive group (the `marker`).

URL

<https://etws.etrade.com/accounts/rest/{accountId}/transactions>

HTTP Method: GET

Request Parameters

The only required parameter is the account ID. By default the API returns a maximum of 50 transactions for the specified account, going back thirty days, and all transaction types are included.

The request parameters and transaction types are shown below, followed by example queries.

Request parameters

Parameter	Type	Required?	Description
accountId	path	required	Numeric account ID. This is the only required parameter.
group	path	optional	Major groupings of the transaction types defined in the Transaction types table below. Possible values are: DEPOSITS, WITHDRAWALS, TRADES.
assetType	path	conditional	Only allowed if <code>group</code> is TRADES. Possible values are: EQ (equities), OPTN (options), MMF (money market funds), MF (mutual funds), BOND (bonds). To retrieve all types, use ALL or omit this parameter.

transactionType	path	optional	Transaction type(s) to include, e.g., check, deposit, fee, dividend, etc. A list of types is provided in the Transaction types table below. If a <code>group</code> is specified, this can be a comma-separated list of valid types for the group. Otherwise, only one type is accepted.
tickerSymbol	path	conditional	Only allowed if <code>group</code> is TRADES. A single market symbol, e.g., GOOG.
startDate	string	optional	The earliest date to include in the date range, formatted as MMDDYYYY. History is available for two years.
endDate	string	optional	The latest date to include in the date range, formatted as MMDDYYYY.
count	integer	optional	Number of transactions to return in the response. If not specified, defaults to 50. Used for paging as described in the Notes below.
marker	integer	optional	Specifies the desired starting point of the set of items to return. Used for paging as described in the Notes below.

Transaction types

Most transaction types are members of groups: trades (T), deposits (D), and withdrawals (W), as shown in the table below. Exceptions are corporate actions, currency exchange transactions, and sweep deposits.

Type	Group	Description
assignment	T	A balance adjustment as the result of another party exercising an option. For a definition of assignments, refer to the E*TRADE online glossary .
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	A balance adjustment as the result of exercising an option
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
pos	D, W	Point of sale debit

sweep		Sweep deposit
transfer	D, W	Cash transfer (in or out)
wire	D, W	Cash wire (in or out)

Sample Requests

Below are some examples of queries to demonstrate the API syntax. The path parameters, if used, appear in the order shown here:

```
https://etws.etrade.com/accounts/rest/{accountId}/transactions/{Group}/{AssetType}/{TransactionType}/{TickerSymbol}.json
```

List all deposits for the last 30 days, up to 50 transactions:

```
https://etws.etrade.com/accounts/rest/83405188/transactions/DEPOSITS
```

All deposits for the year 2011, up to 25 transactions:

```
https://etws.etrade.com/accounts/rest/83405188/transactions/DEPOSITS?startDate=01012011&endDate=12312011&count=25
```

The next 25 deposit transactions (URL received as part of preceding response):

```
https://etws.etrade.com/accounts/rest/83405188/transactions/DEPOSITS.json?startDate=01012011&endDate=12212011&count=25&marker=345678
```

From the withdrawals group, just the ATM fees and service fees:

```
https://etws.etrade.com/accounts/rest/83405188/transactions/WITHDRAWALS/ATM,fee.json
```

From the trades group, all transactions for Google:

```
https://etws.etrade.com/accounts/rest/83405188/transactions/TRADES/OPTN/ALL/GOOG
```

Just the assignment and exercise transactions for Google options:

```
https://etws.etrade.com/accounts/rest/83405188/transactions/TRADES/OPTN/ASSIGNMENT,EXERCISE/GOOG
```

All checking deposits and withdrawals:

<https://etws.etrade.com/accounts/rest/83405188/transactions/TRADES/EQ/>

Response Properties

Property	Type	Description
count	integer	Number of transactions returned in this response
next	string	URL to fetch the next set of transactions ("page") in this series, if there is one. Empty if there are no more transactions to retrieve.
transaction	complex	Container for the elements of a transaction, shown next. The number of these transaction containers is reflected in the <code>count</code> . Appears if there is at least one transaction that meets the requirements.
transactionId	long	Numeric transaction ID
transactionDate	long	Date and time in epoch time
transactionShortDesc	string	Brief description of transaction type, e.g., "Adjustment", "Interest", "Transfer", "Wire", "Contribution", "Fee", "Option Assignment", "Option Expiration", "Option Exercise", "Bought", "Bought To Open", "Sold", etc.
description	string	One-line description of transaction
amount	double	Amount of transaction in dollars, or 0.00 if not applicable
details	string	URL to request the details of this transaction, as described in the Get Transaction Details API

Sample response - XML

```
<transactions>
  <accountId>83405188</accountId>
  <count>2</count>
  <next></next>
  <transaction>
    <transactionId>345678</transactionId>
    <transactionDate>1266519026</transactionDate>
    <transactionShortDesc>Bought</transactionShortDesc>
    <description>10 of AAPL @ $3.00 (Order #141)</description>
    <amount>-39.99</amount>
    <details>
      https://etws.etrade.com/accounts/sandbox/rest/83405188/transactions/345678
    </details>
  </transaction>
  <transaction>
    <transactionId>345678</transactionId>
    <transactionDate>1266519024</transactionDate>
    <transactionShortDesc>Sold</transactionShortDesc>
    <description>1 of KO @ $100.00 (Order #142)</description>
    <amount>90.00</amount>
    <details>
      https://etws.etrade.com/accounts/sandbox/rest/83405188/transactions/345678
    </details>
  </transaction>
</transactions>
```

```
</transactions>
```

Sample response - JSON

```
{
  "transactions": {
    "accountId": "83405188",
    "count": "2",
    "transaction": [
      {
        "transactionId": "345678",
        "transactionDate": "1266519026",
        "transactionShortDesc": "Bought",
        "description": "10 of AAPL @ $3.00 (Order #141)",
        "amount": "-39.99",
        "details":
          "https://etws.etrade.com/accounts/sandbox/rest/83405188/transactions/345678"
      },
      {
        "transactionId": "345678",
        "transactionDate": "1266519024",
        "transactionShortDesc": "Sold",
        "description": "1 of KO @ $100.00 (Order #142)",
        "amount": "90.00",
        "details":
          "https://etws.etrade.com/accounts/sandbox/rest/83405188/transactions/345678"
      }
    ]
  }
}
```

Notes

- To page through a large number of items, use the `count` property to specify how many items to return in a group (the default is 50), and the `marker` property to specify the starting point. A request with no `count` and no `marker` retrieves the most recent 50 transactions. The `next` URL provided in the response retrieves the next set of transactions in the series, using the transaction ID of the last transaction as the `marker`. To page through all the items, repeat the request with the `marker` from each previous response until you receive a response with a `count` less than the one you specified, indicating that there are no more items.
- The API does not explicitly provide for bi-directional paging. Your application can support paging backward and forward either by saving and re-using markers within the series (that is, re-issuing the requests for pages that are earlier in the series), or saving the items returned and re-displaying them from the local copy if desired.

Sample use cases

Some possible use cases and workflows are described below.

Purpose	Workflow	Related APIs
Transaction history	For a selected account, display basic account info, current balances, and a paged history of transactions for the account. Allow the user to filter the history with criteria such as date range, symbol, or transaction type.	Get Transaction History, List Accounts, Get Account Balance
Transaction details	When the user selects a transaction from a list of transactions, the details of the transaction are displayed along with supporting information such as basic account information and a description of any securities involved.	Get Transaction History, Get Transaction Details, List Accounts, Look Up Product

Related APIs

Get Transaction Details, List Accounts, Get Account Balance

Sandbox Samples

Request 1 - Basic listing of transactions

No filter except account ID. A maximum of 10 records is requested.

```
GET
https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions?count=10
```

Response 1

In this response, a `count` of 10 items is returned, plus a `next` URL that can be used for retrieving the next 10 items. Notice that, in the `next` URL, the `marker` is the transaction ID of the last transaction received in this. You can use the provided `next` URL, or construct your own custom URL by using the last transaction ID as a `marker`.

```
<transactions>
  <accountId>83405188</accountId>
  <count>10</count>
  <next>
https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions?count=10&am
p;marker=345678
  </next>
  <transaction>
    <transactionId>345678</transactionId>
    <transactionDate>1266519026</transactionDate>
    <transactionShortDesc>Bought</transactionShortDesc>
    <description>10 of AAPL @ $3.00 (Order #141)</description>
    <amount>-39.99</amount>
    <details>
https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345678
    </details>
  </transaction>
  <transaction>
```

```

    <transactionId>345679</transactionId>
    <transactionDate>1266519024</transactionDate>
    <transactionShortDesc>Sold</transactionShortDesc>
    <description>1 of KO @ $100.00 (Order #142)</description>
    <amount>90.00</amount>
    <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345679
    </details>
  </transaction>
  <transaction>
    <transactionId>345680</transactionId>
    <transactionDate>1266517024</transactionDate>
    <transactionShortDesc>Bought To Open</transactionShortDesc>
    <description>1 IBM Jan 19 &apos;13 $205 Call(IBM) @ $10.10</description>
    <amount>-1012.75</amount>
    <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345680
    </details>
  </transaction>
  <transaction>
    <transactionId>345681</transactionId>
    <transactionDate>1266519026</transactionDate>
    <transactionShortDesc>Bought</transactionShortDesc>
    <description>10 of AAPL bonds</description>
    <amount>-39.99</amount>
    <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345681
    </details>
  </transaction>
  <transaction>
    <transactionId>345682</transactionId>
    <transactionDate>1266519024</transactionDate>
    <transactionShortDesc>Sold</transactionShortDesc>
    <description>1 of KO bonds</description>
    <amount>90.00</amount>
    <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345682
    </details>
  </transaction>
  <transaction>
    <transactionId>345683</transactionId>
    <transactionDate>1266519026</transactionDate>
    <transactionShortDesc>Bought</transactionShortDesc>
    <description>10 of AAPL Mutual funds</description>
    <amount>-39.99</amount>
    <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345683
    </details>
  </transaction>
  <transaction>
    <transactionId>345684</transactionId>
    <transactionDate>1266519024</transactionDate>
    <transactionShortDesc>Sold</transactionShortDesc>
    <description>1 of KO Mutual funds</description>

```

```

    <amount>90.00</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345684
    </details>
  </transaction>
  <transaction>
    <transactionId>345685</transactionId>
    <transactionDate>1266519026</transactionDate>
    <transactionShortDesc>Bought</transactionShortDesc>
    <description>10 of AAPL money market funds</description>
    <amount>-39.99</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345685
    </details>
  </transaction>
  <transaction>
    <transactionId>345686</transactionId>
    <transactionDate>1266519024</transactionDate>
    <transactionShortDesc>Sold</transactionShortDesc>
    <description>1 of KO money market funds</description>
    <amount>90.00</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345686
    </details>
  </transaction>
  <transaction>
    <transactionId>345687</transactionId>
    <transactionDate>1266517024</transactionDate>
    <transactionShortDesc>Option Expiration</transactionShortDesc>
    <description>1 IBM Expiration</description>
    <amount>0</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345687
    </details>
  </transaction>
</transactions>

```

Request 2 - Deposits for single date

```

GET
https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/DEPOSITS?startDate=02182010&endDate=02182010

```

Response 2

In this response, the `count` is 10 and the `next` element is empty. Based on this, we infer that there were 10 deposits on this date.

```

<transactions>
  <accountId>83405188</accountId>
  <count>10</count>

```



```

<next></next>
<transaction>
  <transactionId>123456</transactionId>
  <transactionDate>1266518024</transactionDate>
  <transactionShortDesc>Check</transactionShortDesc>
  <description>Deposited check $100</description>
  <amount>100</amount>
  <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456
  </details>
</transaction>
<transaction>
  <transactionId>123456</transactionId>
  <transactionDate>1266518024</transactionDate>
  <transactionShortDesc>Contribution</transactionShortDesc>
  <description>Deposited $100 contribution</description>
  <amount>100</amount>
  <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456
  </details>
</transaction>
<transaction>
  <transactionId>123456</transactionId>
  <transactionDate>1266518024</transactionDate>
  <transactionShortDesc>Direct Deposit</transactionShortDesc>
  <description>Deposited $100 direct deposit</description>
  <amount>100</amount>
  <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456
  </details>
</transaction>
<transaction>
  <transactionId>123456</transactionId>
  <transactionDate>1266518024</transactionDate>
  <transactionShortDesc>Deposit</transactionShortDesc>
  <description>Deposited $100 deposit</description>
  <amount>100</amount>
  <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456
  </details>
</transaction>
<transaction>
  <transactionId>123456</transactionId>
  <transactionDate>1266518024</transactionDate>
  <transactionShortDesc>Distribution</transactionShortDesc>
  <description>Deposited $100 Distribution</description>
  <amount>100</amount>
  <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456
  </details>
</transaction>
<transaction>
  <transactionId>123456</transactionId>
  <transactionDate>1266518024</transactionDate>
  <transactionShortDesc>Dividend</transactionShortDesc>

```

```

    <description>Deposited $100 Dividend</description>
    <amount>100</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456
    </details>
  </transaction>
  <transaction>
    <transactionId>123456</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Pos</transactionShortDesc>
    <description>Deposited $100 pos</description>
    <amount>100</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456
    </details>
  </transaction>
  <transaction>
    <transactionId>123456</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Interest</transactionShortDesc>
    <description>Deposited $100 Interest</description>
    <amount>100</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456
    </details>
  </transaction>
  <transaction>
    <transactionId>123456</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Transfer</transactionShortDesc>
    <description>TEST</description>
    <amount>100</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456
    </details>
  </transaction>
  <transaction>
    <transactionId>123456</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Wire</transactionShortDesc>
    <description>Deposited $100 Wire</description>
    <amount>100</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456
    </details>
  </transaction>
</transactions>

```

Request 3 - Selected transaction types from a single group

GET

https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/WITHDRAWALS/CHECK,CONTRIBUTION,DIRECT_DEBIT,DISTRIBUTINO,FEE,POS,TRANSFER,WIRE

Response 3

```
<transactions>
  <accountId>83405188</accountId>
  <count>8</count>
  <next></next>
  <transaction>
    <transactionId>234567</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Check</transactionShortDesc>
    <description>Withdrew $100 Check</description>
    <amount>-100</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/234567
    </details>
  </transaction>
  <transaction>
    <transactionId>234567</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Contribution</transactionShortDesc>
    <description>Withdrew $100 contribution</description>
    <amount>-100</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/234567
    </details>
  </transaction>
  <transaction>
    <transactionId>234567</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Withdrawal direct debit</transactionShortDesc>
    <description>Withdrew $100 direct debit</description>
    <amount>100</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/234567
    </details>
  </transaction>
  <transaction>
    <transactionId>234567</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Distribution</transactionShortDesc>
    <description>Withdrew $100 distribution</description>
    <amount>-100</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/234567
    </details>
  </transaction>
  <transaction>
    <transactionId>234567</transactionId>
```

```

    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Fee</transactionShortDesc>
    <description>Subscription:                Pro Dec 11</description>
    <amount>-54.81</amount>
    <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/234567
    </details>
  </transaction>
  <transaction>
    <transactionId>234567</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Pos</transactionShortDesc>
    <description>Withdrew $100 pos</description>
    <amount>-100</amount>
    <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/234567
    </details>
  </transaction>
  <transaction>
    <transactionId>234567</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Transfer</transactionShortDesc>
    <description>TRANSFER TO XXXX0940-1      REFID:140230935;</description>
    <amount>-100</amount>
    <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/234567
    </details>
  </transaction>
  <transaction>
    <transactionId>234567</transactionId>
    <transactionDate>1266518024</transactionDate>
    <transactionShortDesc>Wire</transactionShortDesc>
    <description>Withdrew $100 wire</description>
    <amount>-100</amount>
    <details>
    https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/234567
    </details>
  </transaction>
</transactions>

```

Request 4 - All trades for one asset type and security

This example requests a JSON response by adding ".json" at the end of the path.

```

GET
https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/TRADES/EQ/APL.json

```

Response 4

```
{
```

```

"transactions": {
  "accountId": "83405188",
  "count": "1",
  "transaction": {
    "transactionId": "345678",
    "transactionDate": "1266519026",
    "transactionShortDesc": "Bought",
    "description": "10 of AAPL @ $3.00 (Order #141)",
    "amount": "-39.99",
    "details": "
https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345678
"
  }
}

```

Request 5 - All trades for all asset types for one security

```

GET
https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/TRADES/ALL/
GOOG

```

Response 5

```

<transactions>
  <accountId>83405188</accountId>
  <count>1</count>
  <next></next>
  <transaction>
    <transactionId>345678</transactionId>
    <transactionDate>1266517024</transactionDate>
    <transactionShortDesc>Bought To Open</transactionShortDesc>
    <description>1 IBM Jan 19 &apos;13 $205 Call(IBM) @ $10.10</description>
    <amount>-1012.75</amount>
    <details>
      https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345678
    </details>
  </transaction>
</transactions>

```

Request 6 - Checking account deposits and withdrawals

Lists checking transactions for this account - both deposits and withdrawals. This example requests a JSON response by adding ".json" at the end of the path.

```

GET
https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/CHECK.json?
startDate=01012011&endDate=12312011

```

Response 6

```
{
  "transactions": {
    "accountId": "83405188",
    "count": "2",
    "transaction": [
      {
        "transactionId": "123456",
        "transactionDate": "1266518024",
        "transactionShortDesc": "Check",
        "description": "Deposited check $100",
        "amount": "100",
        "details":
"https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/123456"
      },
      {
        "transactionId": "234567",
        "transactionDate": "1266518024",
        "transactionShortDesc": "Check",
        "description": "Withdrew $100 Check",
        "amount": "-100",
        "details":
"https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/234567"
      }
    ]
  }
}
```

Get Transaction Details

Retrieves the details of a transaction.

Description

This API returns the details of a transaction, based on two required inputs - account ID and transaction ID. Both of these inputs are typically obtained by using the Get Transaction History API.

The same account types and transaction types are supported in both APIs. The details that are returned depend upon the type of transaction, e.g., a stock trade has a `displaySymbol` but a check transaction does not.

URL

<https://etws.etrade.com/accounts/rest/{accountId}/transactions/{transactionId}>

HTTP Method: GET

Request Parameters

Parameter	Type	Required?	Description
accountId	path	required	Numeric account ID
transactionId	path	required	Numeric transaction ID, usually acquired by using the Get Transaction History API.

Sample Request

```
https://etws.etrade.com/accounts/rest/{accountId}/transactions/345678
```

Response Properties

Property	Type	Description
transactionDate	long	Date of the specified transaction
transactionType	string	Type of transaction (deposit, dividend, etc.). Possible values are listed in the Transaction types table below.
userDescription	string	User-defined description
transactionDescription	string	Transaction description
quantity	decimal	Item count (e.g., share count)
amount	decimal	Total cost of transaction, including commission if any
commission	decimal	Commission amount
price	decimal	Price per item if applicable (e.g., price per share)
productID	complex	Container for identity of the product
symbol	string	Security symbol in internal E*TRADE database format

typeCode	string	The type of security. Possible values are: EQ (equity), INDX (index), MF (mutual fund), FI (fixed income).
category	string	User-defined category
displaySymbol	string	Product symbol in client-facing display format , e.g., "GOOG" or "GOOG Dec 17 2012 \$600.50 CALL"
underlyingProductId	complex	Container for identity of the underlier (if applicable)
symbol	string	Market symbol of the underlier, e.g., GOOG
typeCode	string	The type of security for the underlier. Possible values are: EQ (equity), INDX (index), MF (mutual fund), FI (fixed income).
settlementDate	long	Settlement date in epoch time
settlementCurrency	string	Settlement currency
paymentCurrency	string	Payment currency
accountOrderNo	long	Order number

Transaction types

Most transaction types are members of groups: trades (T), deposits (D), and withdrawals (W), as shown in the table below. Exceptions are corporate actions, currency exchange transactions, and sweep deposits. (This is the same table shown under Get Transaction History.)

Type	Group	Description
assignment	T	A balance adjustment as the result of another party exercising an option. For a definition of assignments, refer to the E*TRADE online glossary .
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	A balance adjustment as the result of exercising an option
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
pos	D, W	Point of sale debit

sweep		Sweep deposit
transfer	D, W	Cash transfer (in or out)
wire	D, W	Cash wire (in or out)

Sample response - XML

```
<transactionDetails>
  <transactionDate>1266518024</transactionDate>
  <transactionType>bought</transactionType>
  <userDescription>reimbursement</userDescription>
  <transactionDescription>Bought 10 AAPL @200</transactionDescription>
  <quantity>10</quantity>
  <amount>2010.00</amount>
  <price>200.00</price>
  <commission>10.00</commission>
  <productId>
    <symbol>AAPL</symbol>
    <typeCode>EQ</typeCode>
  </productId>
  <underlyingProductId/>
  <displaySymbol>AAPL</displaySymbol>
  <accountOrderNo>10</accountOrderNo>
  <settlementCurrency>USD</settlementCurrency>
  <paymentCurrency>USD</paymentCurrency>
  <category></category>
  <settlementDate>1322456400</settlementDate>
</transactionDetails>
```

Sample response - JSON

```
{
  "json.transactionDetails":{
    "transactionDate":1266518024,
    "transactionType":"bought",
    "userDescription":"reimbursement",
    "transactionDescription":"Bought 10 AAPL @200",
    "quantity":10,
    "amount":"2010.00",
    "price":"200.00",
    "commission":"10.00",
    "productId":{
      "symbol":"AAPL",
      "typeCode":"EQ"
    },
    "underlyingProductId":"",
    "displaySymbol":"AAPL",
    "accountOrderNo":10,
    "settlementCurrency":"USD",
    "paymentCurrency":"USD",
    "category":"",
    "settlementDate":1322456400
  }
}
```

```
}  
}
```

Sample use cases

Some possible use cases and workflows are described below.

Purpose	Workflow	Related APIs
Transaction detail display	User selects a transaction from a list of transactions that meet specified criteria (e.g., account, date range, symbol, transaction type). The details of the transaction are displayed along with supporting information such as basic account information and a description of any securities involved.	Get Transaction History, Get Transaction Details, List Accounts, Look Up Product

Related APIs

Get Transaction History, List Accounts, Get Account Balance

Sandbox Samples

Request

```
https://etwssandbox.etrade.com/accounts/sandbox/rest/83405188/transactions/345678
```

Response

```
<transactionDetails>  
  <transactionDate>1266518024</transactionDate>  
  <transactionType>bought</transactionType>  
  <userDescription>reimbursement</userDescription>  
  <transactionDescription>Bought 10 AAPL @200</transactionDescription>  
  <quantity>10</quantity>  
  <amount>2010.00</amount>  
  <price>200.00</price>  
  <commission>10.00</commission>  
  <productId>  
    <symbol>AAPL</symbol>  
    <typeCode>EQ</typeCode>  
  </productId>  
  <underlyingProductId/>  
  <displaySymbol>AAPL</displaySymbol>  
  <accountOrderNo>10</accountOrderNo>  
  <settlementCurrency>USD</settlementCurrency>  
  <paymentCurrency>USD</paymentCurrency>  
  <category></category>  
  <settlementDate>1322456400</settlementDate>  
</transactionDetails>
```

Market

Look Up Product

Looks up an exchange and symbol based on company name and security type.

Description

This API returns a list of securities of a specified type (e.g., equity stock) based on a full or partial match of any part of the company name. For instance, a search for "jones" returns a list of securities associated with "Jones Soda Co", "Stella Jones Inc", and many others. The list contains the company name, the exchange that lists the security, the security type, and the symbol, for as many matches as are found.

The result may include some unexpected matches, because the search includes more than just the display version of the company name. For instance, searching on "etrade" returns securities for "E TRADE" - notice the space in the name.

This API is for searching on the company name, not a security symbol. It's commonly used to look up a symbol based on the company name, e.g., "What is the symbol for Google stock?". To look up company information based on a symbol, or to find detailed information on a security, use the `quote` API.

URL

<https://etws.etrade.com/market/rest/productlookup>

HTTP Method: GET

Request Parameters

Parameter	Type	Required?	Description
company	string	required	Full or partial name of the company. Note that the system extensively abbreviates common words such as "company", "industries", and "systems", and generally skips punctuation (periods, commas, apostrophes, etc.) .
type	enum	required	The type of security. Possible values are: EQ (equity) or MF (mutual fund).

Response Properties

Property	Type	Description
companyName	string	The name of the company
exchange	string	The exchange that lists the company
securityType	string	The type of security. Possible values are EQ (equity) and MF (mutual fund).

symbol	string	The market symbol for this security
--------	--------	-------------------------------------

Sample Request

```
GET https://etws.etrade.com/market/rest/productlookup?company=cisco&type=EQ
```

Sample response - XML

```
<productLookupResponse>
  <productList>
    <companyName>CISCO SYS INC COM</companyName>
    <exchange>NASDAQ NM</exchange>
    <securityType>EQ</securityType>
    <symbol>CSCO</symbol>
  </productList>
  <productList>
    <companyName>FRANCISCO INDS INC COM</companyName>
    <exchange>OTC</exchange>
    <securityType>EQ</securityType>
    <symbol>FRAN</symbol>
  </productList>
  <productList>
    <companyName>SAN FRANCISCO CO</companyName>
    <exchange>OTC</exchange>
    <securityType>EQ</securityType>
    <symbol>SFHC</symbol>
  </productList>
</productLookupResponse>
```

Sample response - JSON

```
{
  "productLookupResponse": {
    "productList": [
      {
        "companyName": "CISCO SYS INC COM",
        "exchange": "NASDAQ NM",
        "securityType": "EQ",
        "symbol": "CSCO"
      },
      {
        "companyName": "FRANCISCO INDS INC COM",
        "exchange": "OTC",
        "securityType": "EQ",
        "symbol": "FRAN"
      },
      {
        "companyName": "SAN FRANCISCO CO",
        "exchange": "OTC",
        "securityType": "EQ",
        "symbol": "SFHC"
      }
    ]
  }
}
```

```

    }
  }
}

```

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Look Up Product	Based on user input, display a list of matching company names, security types, and market symbols. User can select a desired symbol to retrieve, for instance, a current market value, fundamental display, and streaming updates.	Look Up Product, Get Quote, Streaming API

Related APIs

Get Quote, Get Option Chains

Sandbox Samples

The following shows a typical request and response in the sandbox environment.

Request

```

GET
https://etwssandbox.etrade.com/market/sandbox/rest/productlookup?company=fi&type=EQ

```

Response

```

<ProductLookupResponse>
  <Product>
    <companyName>E TRADE FINANCIAL CORP COM</companyName>
    <exchange>NASDAQ NM</exchange>
    <securityType>EQ</securityType>
    <symbol>ETFC</symbol>
  </Product>
  <Product>
    <companyName>FT UT 1888DFIETFCM</companyName>
    <exchange>OTCBB</exchange>
    <securityType>EQ</securityType>
    <symbol>FDFXNX</symbol>
  </Product>
</ProductLookupResponse>

```

Get Quote

Returns detailed market information for securities.

Description

This API returns detailed quote information for one or more specified securities. An optional flag specifies one of five pre-configured field sets to return: fundamentals, intraday activity, options, a 52-week display, or all available details (the default). Developers can select the response that most closely fits their needs to minimize data size, processing time, and network traffic.

Version v0 of the E*TRADE API supports real-time market data (as opposed to delayed data, available in some other contexts). Users who have not yet signed the real-time market data agreement on the E*TRADE website receive an error message when trying to access market data with this API.

URL

<https://etws.etrade.com/market/rest/quote/{symbol, symbol...}>

HTTP Method: GET

Request Parameters

Parameter	Type	Required?	Description
symbol	path	required	One or more (comma-separated) symbols for equities or options, up to a maximum of 25. Symbols for equities are simple, e.g., GOOG. Symbols for options are more complex, consisting of six elements separated by colons, in this format: <i>underlier:year:month:day:optionType:strikePrice.</i>
detailFlag	enum	optional	Optional parameter specifying which details to return in the response. The field set for each possible value is listed in separate tables below. The possible values are: <ul style="list-style-type: none">• FUNDAMENTAL - Instrument fundamentals and latest price• INTRADAY - Performance for the current or most recent trading day• OPTIONS - Information on a given option offering• WEEK52 - 52-week high and low (highest high and lowest low)• ALL (default) - All of the above information and more

Response Properties (detailFlag = "ALL")

Property	Type	Description
adjNonAdjFlag	boolean	Indicates whether an option has been adjusted due to a corporate action (e.g. a dividend or stock split). Possible values are TRUE, FALSE.
annualDividend	double	Cash amount paid per share over the past year
ask	double	The current ask price for a security

askExchange	string	Code for the exchange reporting the ask price. Exchange codes are listed in a separate table below.
askSize	long	Number of shares or contracts offered by a broker/dealer at the ask price
askTime	string	Time of the ask, e.g., "15:15:43 EDT 03-15-2011"
beta	double	A measure of a stock's volatility relative to the primary market index
bid	double	Current bid price for a security
bidExchange	string	Code for the exchange reporting the bid price. Exchange codes are listed in a separate table below.
bidSize	long	Number of shares or contracts offered at the bid price
bidTime	string	Time of the bid, e.g., "15:15:43 EDT 03-15-2011"
chgClose	double	Dollar change of the last price from the previous close
chgClosePrcn	double	Percentage change of the last price from the previous close
companyName	string	Name of the company or mutual fund. (Shows up to 40 characters.)
daysToExpiration	long	Number of days before the option expires
dirLast	string	Direction of movement, i.e., whether the current price is higher or lower than the price of the most recent trade
dividend	double	Cash amount per share of the latest dividend
eps	double	Earnings per share on a rolling basis. (Stocks only.)
estEarnings	double	Projected earnings per share for the next fiscal year. (Stocks only.)
exDivDate	string	Date on which shareholders were entitled to receive the latest dividend, in epoch time.
exchgLastTrade	string	Code for the exchange of the last trade. Exchange codes are listed in a separate table below.
fsi	string	Financial Status Indicator - indicates whether a Nasdaqlisted issuer has failed to submit its regulatory filings on a timely basis, failed to meet continuing listing standards, and/or filed for bankruptcy. Codes are: <ul style="list-style-type: none"> • D - Deficient • E - Delinquent • Q - Bankrupt • N - Normal • G - Deficient and Bankrupt • H - Deficient and Delinquent • J - Delinquent and Bankrupt • K - Deficient, Delinquent, and Bankrupt
high	double	Highest price at which a security has traded during the current day
high52	double	Highest price at which a security has traded during the past year (52 weeks). For options, this value is the lifetime high.
highAsk	double	Highest ask price for the current trading day
highBid	double	Highest bid price for the current trading day
lastTrade	double	Price of the most recent trade in a security
low	double	Lowest price at which a security has traded during the current day
low52	double	Lowest price at which a security has traded during the past year (52 weeks). For options, this value is the lifetime low.

lowAsk	double	Lowest ask price for the current trading day
lowBid	double	Lowest bid price for the current trading day
numTrades	long	Number of transactions that involve buying a security from another entity
open	double	Price of a security at the current day's market open
openInterest	long	Total number of options or futures contracts that are not closed or delivered on a particular day
optionStyle	string	Specifies how the contract treats the expiration date. Possible values are "European" (options can be exercised only on the expiration date) or "American" (options can be exercised any time before it expires).
optionUnderlier	string	Symbol for the underlier. (Options only.)
prevClose	double	Official price at the close of the previous trading day
prevDayVolume	long	Final volume from the previous market session
primaryExchange	string	Exchange code of the primary listing exchange for this instrument. (Exchange codes are listed in a separate table below.)
symbolDesc	string	Description of the security - e.g., the company name or option description.
todayClosed	double	Price at the close of the regular trading session for the current day.
totalVolume	long	Total number of shares or contracts exchanging hands.
upc	long	Uniform Practice Code - identifies specific FINRA advisories detailing unusual circumstances (e.g., extremely large dividends, when-issued settlement dates, and worthless securities).
volume10Day	long	Ten-day average trading volume for the security
dateTime	string	Time of this quote, e.g., "15:15:43 EDT 03-15-2011"
product	complex	Container for product information
symbol	string	Market trading symbol for the stock
type	string	Type of security. Possible values are: EQ and OPTN.
exchange	string	Exchange that lists this company. (Exchange codes are listed in a separate table below.)

Response Properties (detailFlag = "FUNDAMENTAL")

Property	Type	Description
beta	double	A measure of a stock's volatility relative to the primary market index
eps	double	Earnings per share on a rolling basis. (Applies to stocks only.)
estEarnings	double	Projected earnings per share for the next fiscal year. (Applies to stocks only.)
high52	double	The highest price at which a security has traded during the past year (52 weeks). For options, this value is the lifetime high.
lastTrade	double	The price of the most recent trade in a security
low52	double	The lowest price at which a security has traded during the past year (52 weeks). For options, this value is the lifetime low.
symbolDesc	string	A description of the security, such as company name or option description
volume10Day	long	Ten-day average trading volume for the security
dateTime	long	The time of this quote, e.g., "15:15:43 EDT 03-15-2011"
product	complex	Container for symbol information

symbol	string	The market trading symbol for the stock
type	string	The type of security. Possible values are: EQ and OPTN.
exchange	string	The exchange that lists this company. (Exchange codes are listed in a separate table below.)

Response Properties (detailFlag = "INTRADAY")

Property	Type	Description
ask	double	The current ask price for a security
bid	double	The current bid price for a security
chgClose	double	The dollar change of the last price from the previous close
chgClosePrcn	double	The percentage change of the last price from the previous close
high	double	The highest price at which a security has traded during the current day
lastTrade	double	The price of the most recent trade in a security
low	double	The lowest price at which a security has traded during the current day
totalVolume	long	Total number of shares or contracts exchanging hands
dateTime	long	The time of this quote, e.g., "15:15:43 EDT 03-15-2011"
product	complex	Container for symbol information
symbol	string	The market trading symbol for the stock
type	string	The type of security. Possible values are: EQ and OPTN.
exchange	string	The exchange that lists this company. (Exchange codes are listed in a separate table below.)

Response Properties (detailFlag = "OPTIONS")

Property	Type	Description
ask	double	The current ask price for a security
askSize	long	The number of shares or contracts offered by a broker/dealer at the ask price
bid	double	The current bid price for a security
bidSize	long	The number of shares or contracts offered at the bid price
daysToExpiration	long	Number of days before the option expires
lastTrade	double	The price of the most recent trade in a security
openInterest	long	The total number of options or futures contracts that are not closed or delivered on a particular day
dateTime	string	The time of this quote, e.g., "15:15:43 EDT 03-15-2011"
product	complex	Container for symbol information
symbol	string	The market trading symbol for the stock
type	string	The type of security. Possible values are: EQ and OPTN.
exchange	string	The exchange that lists this company. (Exchange codes are listed in a separate table below.)

Response Properties (detailFlag = "WEEK_52")

Property	Type	Description
annualDividend	double	The cash dividend paid per share over the past year
high	double	The highest price at which a security has traded during the current day
lastTrade	double	The price of the most recent trade in a security
low	double	The lowest price at which a security has traded during the current day
prevClose	double	The official price at the close on the previous trading day
symbolDesc	string	A description of the security - e.g., company name or option description
totalVolume	long	Total number of shares or contracts exchanging hands
dateTime	long	The time of this quote, e.g., "15:15:43 EDT 03-15-2011"
product	complex	Container for symbol information
symbol	string	The market trading symbol for the stock in the quote
type	string	The type of security. Possible values are: EQ and OPTN.
exchange	string	The exchange that lists this company. (Exchange codes are listed in a separate table below.)

Sample Request

```
GET https://etrade.com/market/rest/quote/GOOG?detailFlag=ALL
```

Sample response - XML (detailFlag = "ALL")

```
<quoteResponse>
  <quoteData>
    <all>
      <adjNonAdjFlag>false</adjNonAdjFlag>
      <annualDividend>0</annualDividend>
      <ask>603.94</ask>
      <askExchange>Q</askExchange>
      <askSize>100</askSize>
      <askTime>14:35:42 EST 01-04-2011</askTime>
      <bid>603.75</bid>
      <bidExchange>P</bidExchange>
      <bidSize>100</bidSize>
      <bidTime>14:35:42 EST 01-04-2011</bidTime>
      <chgClose>-0.4900000000000091</chgClose>
      <chgClosePrcn>-0.08</chgClosePrcn>
      <companyName>GOOGLE INC</companyName>
      <daysToExpiration>0</daysToExpiration>
      <dirLast>U</dirLast>
      <dividend>0</dividend>
      <eps>24.63</eps>
      <estEarnings>33.642</estEarnings>
      <exDivDate></exDivDate>
      <exchgLastTrade></exchgLastTrade>
      <fsi>N</fsi>
      <high>606.18</high>
      <high52>630.85</high52>
```

```

    <highAsk>640</highAsk>
    <highBid>606.79</highBid>
    <lastTrade>603.86</lastTrade>
    <low>600.12</low>
    <low52>433.63</low52>
    <lowAsk>600.12</lowAsk>
    <lowBid>449.47</lowBid>
    <numTrades>10590</numTrades>
    <open>606.4</open>
    <openInterest>0</openInterest>
    <optionStyle></optionStyle>
    <optionUnderlier></optionUnderlier>
    <prevClose>604.35</prevClose>
    <prevDayVolume>2366286</prevDayVolume>
    <primaryExchange>Q </primaryExchange>
    <symbolDesc>GOOGLE INC</symbolDesc>
    <todayClose>0</todayClose>
    <totalVolume>1412611</totalVolume>
    <upc>0</upc>
    <volume10Day>1436450</volume10Day>
  </all>
  <dateTime>14:35:30 EST 01-04-2011</dateTime>
  <product>
    <symbol>GOOG</symbol>
    <type>EQ</type>
    <exchange>Q</exchange>
  </product>
</quoteData>
</quoteResponse>

```

Sample response - JSON (detailFlag = "ALL")

```

{
  "quoteResponse":{
    "quoteData":{
      "all":{
        "adjNonAdjFlag":false,
        "annualDividend":0,
        "ask":603.94,
        "askExchange":"Q",
        "askSize":100,
        "askTime":"14:35:42 EST 01-04-2011",
        "bid":603.75,
        "bidExchange":"P",
        "bidSize":100,
        "bidTime":"14:35:42 EST 01-04-2011",
        "chgClose":-0.49000000000000091,
        "chgClosePrcn":-0.08,
        "companyName":"GOOGLE INC",
        "daysToExpiration":0,
        "dirLast":"U",
        "dividend":0,
        "eps":24.63,

```

```

        "estEarnings":33.642,
        "exDivDate":"",
        "exchgLastTrade":"",
        "fsi":"N",
        "high":606.18,
        "high52":630.85,
        "highAsk":640,
        "highBid":606.79,
        "lastTrade":603.86,
        "low":600.12,
        "low52":433.63,
        "lowAsk":600.12,
        "lowBid":449.47,
        "numTrades":10590,
        "open":606.4,
        "openInterest":0,
        "optionStyle":"",
        "optionUnderlier":"",
        "prevClose":604.35,
        "prevDayVolume":2366286,
        "primaryExchange":"Q ",
        "symbolDesc":"GOOGLE INC",
        "todayClose":0,
        "totalVolume":1412611,
        "upc":0,
        "volume10Day":1436450
    },
    "dateTime":"14:35:30 EST 01-04-2011",
    "product":{
        "symbol":"GOOG",
        "type":"EQ",
        "exchange":"Q"
    }
}
}
}
}

```

Sample response - XML (detailFlag = "FUNDAMENTAL")

```

<QuoteResponse>
  <QuoteData>
    <dateTime>15:15:43 EDT 03-15-2011</dateTime>
    <fundamental>
      <companyName>GOOGLE INC</companyName>
      <eps>24.63</eps>
      <estEarnings>34.524</estEarnings>
      <high52>642.96</high52>
      <lastTrade>570.92</lastTrade>
      <low52>433.63</low52>
      <symbolDesc>GOOGLE INC</symbolDesc>
      <volume10Day>2718247</volume10Day>
    </fundamental>
    <product>
      <symbol>GOOG</symbol>
    </product>
  </QuoteData>
</QuoteResponse>

```

```

        <type>EQ</type>
        <exchange>Q</exchange>
    </product>
</QuoteData>
<QuoteData>
    <dateTime>12:15:58 EST 02-24-2010</dateTime>
    <fundamental>
        <companyName>MICROSOFT CORP</companyName>
        <eps>1.81</eps>
        <estEarnings>1.811</estEarnings>
        <high52>31.5</high52>
        <lastTrade>28.71</lastTrade>
        <low52>14.87</low52>
        <symbolDesc>MICROSOFT CORP</symbolDesc>
        <volume10Day>56676460</volume10Day>
    </fundamental>
    <product>
        <symbol>MSFT</symbol>
        <type>EQ</type>
        <exchange>Q</exchange>
    </product>
</QuoteData>
</QuoteResponse>

```

Sample response - JSON (detailFlag = "FUNDAMENTAL")

```

{
  "QuoteResponse": {
    "QuoteData": [
      {
        "dateTime": "15:15:43 EDT 03-15-2011",
        "fundamental": {
          "companyName": "GOOGLE INC",
          "eps": "24.63",
          "estEarnings": "34.524",
          "high52": "642.96",
          "lastTrade": "570.92",
          "low52": "433.63",
          "symbolDesc": "GOOGLE INC",
          "volume10Day": "2718247"
        },
        "product": {
          "symbol": "GOOG",
          "type": "EQ",
          "exchange": "Q"
        }
      },
      {
        "dateTime": "12:15:58 EST 02-24-2010",
        "fundamental": {
          "companyName": "MICROSOFT CORP",
          "eps": "1.81",
          "estEarnings": "1.811",
          "high52": "31.5",

```

```

        "lastTrade": "28.71",
        "low52": "14.87",
        "symbolDesc": "MICROSOFT CORP",
        "volume10Day": "56676460"
    },
    "product": {
        "symbol": "MSFT",
        "type": "EQ",
        "exchange": "Q"
    }
}
]
}
}

```

Sample response - XML (detailFlag = "INTRADAY")

```

<quoteResponse>
  <quoteData>
    <dateTime>14:52:03 EST 01-04-2011</dateTime>
    <intraday>
      <ask>602.56</ask>
      <bid>602.45</bid>
      <chgClose>-1.89999999999999773</chgClose>
      <chgClosePrcn>-0.31</chgClosePrcn>
      <companyName>GOOGLE INC</companyName>
      <high>606.18</high>
      <lastTrade>602.45</lastTrade>
      <low>600.12</low>
      <totalVolume>1465065</totalVolume>
    </intraday>
    <product>
      <symbol>GOOG</symbol>
      <type>EQ</type>
      <exchange>Q</exchange>
    </product>
  </quoteData>
</quoteResponse>

```

Sample response - JSON (detailFlag = "INTRADAY")

```

{
  "quoteResponse": {
    "quoteData": {
      "dateTime": "14:52:03 EST 01-04-2011",
      "intraday": {
        "ask": 602.56,
        "bid": 602.45,
        "chgClose": -1.89999999999999773,
        "chgClosePrcn": -0.31,
        "companyName": "GOOGLE INC",
        "high": 606.18,
        "lastTrade": 602.45,

```

```

        "low":600.12,
        "totalVolume":1465065
    },
    "product":{
        "symbol":"GOOG",
        "type":"EQ",
        "exchange":"Q"
    }
}
}
}

```

Sample response - XML (detailFlag = "OPTIONS")

```

<quoteResponse>
  <quoteData>
    <dateTime>14:52:52 EST 01-04-2011</dateTime>
    <option>
      <ask>602.73</ask>
      <askSize>100</askSize>
      <bid>602.57</bid>
      <bidSize>300</bidSize>
      <companyName>GOOGLE INC</companyName>
      <daysToExpiration>0</daysToExpiration>
      <lastTrade>602.7299</lastTrade>
      <openInterest>0</openInterest>
    </option>
    <product>
      <symbol>GOOG</symbol>
      <type>EQ</type>
      <exchange>Q</exchange>
    </product>
  </quoteData>
</quoteResponse>

```

Sample response - JSON (detailFlag = "OPTIONS")

```

{
  "quoteResponse":{
    "quoteData":{
      "dateTime":"14:52:52 EST 01-04-2011",
      "option":{
        "ask":602.73,
        "askSize":100,
        "bid":602.57,
        "bidSize":300,
        "companyName":"GOOGLE INC",
        "daysToExpiration":0,
        "lastTrade":602.7299,
        "openInterest":0
      },
      "product":{
        "symbol":"GOOG",

```

```

        "type":"EQ",
        "exchange":"Q"
    }
}
}
}

```

Sample response - XML (detailFlag = "WEEK_52")

```

<quoteResponse>
  <quoteData>
    <dateTime>14:54:41 EST 01-04-2011</dateTime>
    <product>
      <symbol>GOOG</symbol>
      <type>EQ</type>
      <exchange>Q</exchange>
    </product>
    <week52>
      <annualDividend>0</annualDividend>
      <companyName>GOOGLE INC</companyName>
      <high52>630.85</high52>
      <lastTrade>602.6</lastTrade>
      <low52>433.63</low52>
      <perf12Months>97</perf12Months>
      <prevClose>604.35</prevClose>
      <symbolDesc>GOOGLE INC</symbolDesc>
      <totalVolume>1469699</totalVolume>
    </week52>
  </quoteData>
</quoteResponse>

```

Sample response - JSON (detailFlag = "WEEK_52")

```

{
  "quoteResponse":{
    "quoteData":{
      "dateTime":"14:54:41 EST 01-04-2011",
      "product":{
        "symbol":"GOOG",
        "type":"EQ",
        "exchange":"Q"
      },
      "week52":{
        "annualDividend":0,
        "companyName":"GOOGLE INC",
        "high52":630.85,
        "lastTrade":602.6,
        "low52":433.63,
        "perf12Months":97,
        "prevClose":604.35,
        "symbolDesc":"GOOGLE INC",
        "totalVolume":1469699
      }
    }
  }
}

```



```

    }
  }
}

```

Exchange Codes

Code	Exchange
A	American Stock Exchange
G	Amex Emerging Company Marketplace
AP	Archipelago
Z	BATS
M	Chicago (Midwest) Stock Exchange
CO	Chicago Board Options Exchange
C	Cincinnati Stock Exchange
GP	GovPX Bonds
I	International Securities Exchange (Options)
IS	Island
XT	Market XT
V	Nasdaq Bulletin Board, Trades Only
F	NASDAQ Mutual Fund & Money Market Fund
Q	Nasdaq National Market System (NMS)
B	NASDAQ OMX BX
U	Nasdaq OTC Bulletin Board (Pink Sheet Stocks)
S	Nasdaq Small Cap
T	Nasdaq Trades in Listed Stocks
N	New York Stock Exchange
R	New York Stock Exchange Trade Reporting Facility (TRF)
P	Pacific Stock Exchange
P	Pacific Stock Exchange, Tier I
PT	Pacific Stock Exchange, Tier II
X	Philadelphia Stock Exchange
RD	Red Book

Notes

- This API takes an instrument symbol as input. In some use cases it will be necessary to use the Look Up Product API to find the symbol based on a company name provided by the user.
- The API allows up to 25 symbols to be specified per request. Such a request counts as a single request, not multiple requests, so it does not exceed the rate limit.
- The E*TRADE API v0 supports real-time market data (as opposed to delayed data, available in some other contexts). Users who have not yet signed the real-time market

data agreement on the E*TRADE website receive an error message when trying to access market data with this API.

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Symbol lookup	Use the Get Quote API to look up basic information such as company name or product description for a product. This is used in multiple places within an application, e.g., when hovering over a symbol.	Get Quote
Product display with updating quote	Display a list of stocks for a given company name, and allow a user to select a stock and specify a type of quote display (e.g., fundamentals). Show the quote along with a box that shows dynamically-updating price data.	Look Up Product, Get Quote, Streaming API

Related APIs

Look Up Product, Get Option Chains, Get Option Expire Dates

Sandbox Samples

Below are example sandbox requests and responses for the following common operations:

1. Request all information (the default) for a single security
2. Request all information for multiple securities
3. Request fundamentals for a single security
4. Request intraday for multiple securities
5. Request options for a single security
6. Request 52-week data for a single security

Request 1 - All (single)

```
GET https://etwssandbox.etrade.com/market/sandbox/rest/quote/AAPL
```

Response 1

```
<QuoteResponse>
  <QuoteData>
    <all>
      <adjNonAdjFlag>false</adjNonAdjFlag>
      <annualDividend>0.0</annualDividend>
      <ask>0.0</ask>
```

```

<askExchange>K</askExchange>
<askSize>0</askSize>
<askTime>00:00:00 EDT 03-15-2011</askTime>
<bid>0.0</bid>
<bidExchange>K</bidExchange>
<bidSize>0</bidSize>
<bidTime>00:00:00 EDT 03-15-2011</bidTime>
<chgClose>0.3100000000000023</chgClose>
<chgClosePrcn>0.09</chgClosePrcn>
<companyName>APPLE INC</companyName>
<daysToExpiration>0</daysToExpiration>
<dirLast>D</dirLast>
<dividend>0.0</dividend>
<eps>15.15</eps>
<estEarnings>23.021</estEarnings>
<exDivDate></exDivDate>
<exchgLastTrade>Pacific</exchgLastTrade>
<fsi>N</fsi>
<high>0.0</high>
<high52>364.9</high52>
<highAsk>0.0</highAsk>
<highBid>0.0</highBid>
<lastTrade>353.87</lastTrade>
<low>0.0</low>
<low52>199.25</low52>
<lowAsk>0.0</lowAsk>
<lowBid>0.0</lowBid>
<numTrades>0</numTrades>
<open>0.0</open>
<openInterest>0</openInterest>
<optionStyle></optionStyle>
<optionUnderlier></optionUnderlier>
<prevClose>353.56</prevClose>
<prevDayVolume>15587670</prevDayVolume>
<primaryExchange>Q</primaryExchange>
<symbolDesc>APPLE INC</symbolDesc>
<todayClose>353.87</todayClose>
<totalVolume>0</totalVolume>
<upc>0</upc>
<volume10Day>17085828</volume10Day>
</all>
<dateTime>15:15:54 EDT 03-15-2011</dateTime>
<product>
  <symbol>AAPL</symbol>
  <type>EQ</type>
  <exchange>Q</exchange>
</product>
</QuoteData>
</QuoteResponse>

```

Request 2 - All (multiple)

```
GET https://etwssandbox.etrade.com/market/sandbox/rest/quote/GOOG,AAPL,MSFT
```

Response 2

```
<QuoteResponse>
  <QuoteData>
    <all>
      <adjNonAdjFlag>false</adjNonAdjFlag>
      <annualDividend>0.0</annualDividend>
      <ask>0.0</ask>
      <askExchange>Y</askExchange>
      <askSize>0</askSize>
      <askTime>00:00:00 EDT 03-15-2011</askTime>
      <bid>0.0</bid>
      <bidExchange>Q</bidExchange>
      <bidSize>0</bidSize>
      <bidTime>00:00:00 EDT 03-15-2011</bidTime>
      <chgClose>0.92999999999995</chgClose>
      <chgClosePr-cn>0.16</chgClosePr-cn>
      <companyName>GOOGLE INC</companyName>
      <daysToExpiration>0</daysToExpiration>
      <dirLast>D</dirLast>
      <dividend>0.0</dividend>
      <eps>24.63</eps>
      <estEarnings>34.524</estEarnings>
      <exDivDate></exDivDate>
      <exchgLastTrade>NASDAQ</exchgLastTrade>
      <fsi>N</fsi>
      <high>0.0</high>
      <high52>642.96</high52>
      <highAsk>0.0</highAsk>
      <highBid>0.0</highBid>
      <lastTrade>570.92</lastTrade>
      <low>0.0</low>
      <low52>433.63</low52>
      <lowAsk>0.0</lowAsk>
      <lowBid>0.0</lowBid>
      <numTrades>0</numTrades>
      <open>0.0</open>
      <openInterest>0</openInterest>
      <optionStyle></optionStyle>
      <optionUnderlier></optionUnderlier>
      <prevClose>569.99</prevClose>
      <prevDayVolume>2813969</prevDayVolume>
      <primaryExchange>Q</primaryExchange>
      <symbolDesc>GOOGLE INC</symbolDesc>
      <todayClose>570.92</todayClose>
      <totalVolume>0</totalVolume>
      <upc>0</upc>
      <volume10Day>2718247</volume10Day>
    </all>
  <dateTime>15:15:43 EDT 03-15-2011</dateTime>
```

```

<product>
  <symbol>GOOG</symbol>
  <type>EQ</type>
  <exchange>Q</exchange>
</product>
</QuoteData>
<QuoteData>
  <all>
    <adjNonAdjFlag>false</adjNonAdjFlag>
    <annualDividend>0.0</annualDividend>
    <ask>0.0</ask>
    <askExchange>K</askExchange>
    <askSize>0</askSize>
    <askTime>00:00:00 EDT 03-15-2011</askTime>
    <bid>0.0</bid>
    <bidExchange>K</bidExchange>
    <bidSize>0</bidSize>
    <bidTime>00:00:00 EDT 03-15-2011</bidTime>
    <chgClose>0.3100000000000023</chgClose>
    <chgClosePrctn>0.09</chgClosePrctn>
    <companyName>APPLE INC</companyName>
    <daysToExpiration>0</daysToExpiration>
    <dirLast>D</dirLast>
    <dividend>0.0</dividend>
    <eps>15.15</eps>
    <estEarnings>23.021</estEarnings>
    <exDivDate></exDivDate>
    <exchgLastTrade>Pacific</exchgLastTrade>
    <fsi>N</fsi>
    <high>0.0</high>
    <high52>364.9</high52>
    <highAsk>0.0</highAsk>
    <highBid>0.0</highBid>
    <lastTrade>353.87</lastTrade>
    <low>0.0</low>
    <low52>199.25</low52>
    <lowAsk>0.0</lowAsk>
    <lowBid>0.0</lowBid>
    <numTrades>0</numTrades>
    <open>0.0</open>
    <openInterest>0</openInterest>
    <optionStyle></optionStyle>
    <optionUnderlier></optionUnderlier>
    <prevClose>353.56</prevClose>
    <prevDayVolume>15587670</prevDayVolume>
    <primaryExchange>Q</primaryExchange>
    <symbolDesc>APPLE INC</symbolDesc>
    <todayClose>353.87</todayClose>
    <totalVolume>0</totalVolume>
    <upc>0</upc>
    <volume10Day>17085828</volume10Day>
  </all>
  <dateTime>15:15:54 EDT 03-15-2011</dateTime>
</product>
  <symbol>AAPL</symbol>

```

```

    <type>EQ</type>
    <exchange>Q</exchange>
  </product>
</QuoteData>
<QuoteData>
  <all>
    <adjNonAdjFlag>false</adjNonAdjFlag>
    <annualDividend>0.52</annualDividend>
    <ask>28.71</ask>
    <askExchange>NASDAQ National Market Sys (NMS)</askExchange>
    <askSize>15600</askSize>
    <askTime>12:14:22 EST 02-24-2010</askTime>
    <bid>28.7</bid>
    <bidExchange></bidExchange>
    <bidSize>15600</bidSize>
    <bidTime>12:14:22 EST 02-24-2010</bidTime>
    <chgClose>-0.3400000000000001</chgClose>
    <chgClosePrcn>-15.11</chgClosePrcn>
    <companyName>MICROSOFT CORP</companyName>
    <daysToExpiration>0</daysToExpiration>
    <dirLast>U</dirLast>
    <dividend>0.13</dividend>
    <eps>1.81</eps>
    <estEarnings>1.811</estEarnings>
    <exDivDate>11/17/2009</exDivDate>
    <exchgLastTrade>Pacific</exchgLastTrade>
    <fsi>N</fsi>
    <high>28.77</high>
    <high52>31.5</high52>
    <highAsk>28.77</highAsk>
    <highBid>28.76</highBid>
    <lastTrade>28.705</lastTrade>
    <low>28.38</low>
    <low52>14.87</low52>
    <lowAsk>28.39</lowAsk>
    <lowBid>28.0</lowBid>
    <numTrades>61376</numTrades>
    <open>28.52</open>
    <openInterest>0</openInterest>
    <optionStyle></optionStyle>
    <optionUnderlier></optionUnderlier>
    <prevClose>28.73</prevClose>
    <prevDayVolume>75648887</prevDayVolume>
    <primaryExchange>Q</primaryExchange>
    <symbolDesc>MICROSOFT CORP</symbolDesc>
    <todayClose>0.0</todayClose>
    <totalVolume>22403011</totalVolume>
    <upc>0</upc>
    <volume10Day>56676460</volume10Day>
  </all>
  <dateTime>12:14:22 EST 02-24-2010</dateTime>
  <product>
    <symbol>MSFT</symbol>
    <type>EQ</type>
    <exchange>Q</exchange>
  </product>

```

```
</product>
</QuoteData>
</QuoteResponse>
```

Request 3 - Fundamentals (single)

```
GET
https://etwssandbox.etrade.com/market/sandbox/rest/quote/MSFT?detailFlag=FUNDAMENTAL
```

Response 3

```
<QuoteResponse>
  <QuoteData>
    <dateTime>12:15:58 EST 02-24-2010</dateTime>
    <fundamental>
      <companyName>MICROSOFT CORP</companyName>
      <eps>1.81</eps>
      <estEarnings>1.811</estEarnings>
      <high52>31.5</high52>
      <lastTrade>28.71</lastTrade>
      <low52>14.87</low52>
      <symbolDesc>MICROSOFT CORP</symbolDesc>
      <volume10Day>56676460</volume10Day>
    </fundamental>
    <product>
      <symbol>MSFT</symbol>
      <type>EQ</type>
      <exchange>Q</exchange>
    </product>
  </QuoteData>
</QuoteResponse>
```

Request 4 - Intraday (multiple)

```
GET
https://etwssandbox.etrade.com/market/sandbox/rest/quote/GOOG,AAPL,MSFT?detailFlag=INTRADAY
```

Response 4

```
<QuoteResponse>
  <QuoteData>
    <dateTime>15:15:43 EDT 03-15-2011</dateTime>
    <intraday>
      <ask>0.0</ask>
      <bid>0.0</bid>
      <chgClose>0.92999999999995</chgClose>
      <chgClosePrcn>0.16</chgClosePrcn>
    </intraday>
  </QuoteData>
</QuoteResponse>
```

```

    <companyName>GOOGLE INC</companyName>
    <high>0.0</high>
    <lastTrade>570.92</lastTrade>
    <low>0.0</low>
    <totalVolume>0</totalVolume>
  </intraday>
  <product>
    <symbol>GOOG</symbol>
    <type>EQ</type>
    <exchange>Q</exchange>
  </product>
</QuoteData>
<QuoteData>
  <dateTime>15:15:54 EDT 03-15-2011</dateTime>
  <intraday>
    <ask>0.0</ask>
    <bid>0.0</bid>
    <chgClose>0.3100000000000023</chgClose>
    <chgClosePrcn>0.09</chgClosePrcn>
    <companyName>APPLE INC</companyName>
    <high>0.0</high>
    <lastTrade>353.87</lastTrade>
    <low>0.0</low>
    <totalVolume>0</totalVolume>
  </intraday>
  <product>
    <symbol>AAPL</symbol>
    <type>EQ</type>
    <exchange>Q</exchange>
  </product>
</QuoteData>
<QuoteData>
  <dateTime>12:19:32 EST 02-24-2010</dateTime>
  <intraday>
    <ask>28.74</ask>
    <bid>28.73</bid>
    <chgClose>0.00999999999999801</chgClose>
    <chgClosePrcn>0.03</chgClosePrcn>
    <companyName>MICROSOFT CORP</companyName>
    <high>28.77</high>
    <lastTrade>28.74</lastTrade>
    <low>28.38</low>
    <totalVolume>22756597</totalVolume>
  </intraday>
  <product>
    <symbol>MSFT</symbol>
    <type>EQ</type>
    <exchange>Q</exchange>
  </product>
</QuoteData>
</QuoteResponse>

```


Request 5 - Options (single)

```
GET https://etwssandbox.etrade.com/market/sandbox/rest/quote/MSFT?detailFlag=OPTIONS
```

Response 5

```
<QuoteResponse>
  <QuoteData>
    <dateTime>12:18:18 EST 02-24-2010</dateTime>
    <option>
      <ask>28.72</ask>
      <askSize>11200</askSize>
      <bid>28.71</bid>
      <bidSize>19100</bidSize>
      <companyName>MICROSOFT CORP</companyName>
      <daysToExpiration>0</daysToExpiration>
      <lastTrade>28.72</lastTrade>
      <openInterest>0</openInterest>
    </option>
    <product>
      <symbol>MSFT</symbol>
      <type>EQ</type>
      <exchange>Q</exchange>
    </product>
  </QuoteData>
</QuoteResponse>
```

Request 6 - 52-week (single)

```
GET https://etwssandbox.etrade.com/market/sandbox/rest/quote/MSFT?detailFlag=WEEK_52
```

Response 6

```
<QuoteResponse>
  <QuoteData>
    <dateTime>12:17:30 EST 02-24-2010</dateTime>
    <product>
      <symbol>MSFT</symbol>
      <type>EQ</type>
      <exchange>Q</exchange>
    </product>
    <week52>
      <annualDividend>0.52</annualDividend>
      <companyName>MICROSOFT CORP</companyName>
      <high52>31.5</high52>
      <lastTrade>28.705</lastTrade>
      <low52>14.87</low52>
      <perf12Months>0.0</perf12Months>
      <prevClose>28.73</prevClose>
      <symbolDesc>MICROSOFT CORP</symbolDesc>
      <totalVolume>22434786</totalVolume>
    </week52>
  </QuoteData>
</QuoteResponse>
```

```
</week52>  
</QuoteData>  
</QuoteResponse>
```

Get Option Chains

Returns a list of option chains associated with a specific underlier.

Description

This API returns a list of option chains for a specific underlying instrument. The request must specify an instrument, the month the option expires, and whether you are interested in calls, puts, or both. Values returned include the option pair count and information about each option pair, including the type, call count, symbol, product, date, and strike price.

URL

<https://etws.etrade.com/market/rest/optionchains>

HTTP Method: GET

Request Parameters

Parameter	Type	Required?	Description
chainType	enum	required	The type of option chain. Possible values are: CALL, PUT, or CALLPUT (i.e., both calls and puts).
expirationMonth	integer	required	The month the option will expire
expirationYear	integer	required	The year the option will expire
underlier	string	required	The market symbol for the underlying security
skipAdjusted	boolean	optional	Specifies whether to show (TRUE) or not show (FALSE) adjusted options, i.e., options that have undergone a change resulting in a modification of the option contract. Default value is TRUE.

Response Properties

Property	Type	Description
optionPairCount	integer	The number of option pairs returned
optionPairs	complex	Container for an option pair. There will be one of these for each option pair in the response.
callCount (or putCount)	integer	The number of call or put objects in this container
pairType	string	Determines whether the response will contain calls, puts, or both. Possible values are: CALLONLY, PUTONLY, or CALLPUT.
call (or put)	complex	Container for a call or put. There are typically more than one of these in a response - check callCount or putCount for the count.
rootSymbol	string	The underlier for the option as originally contracted, e.g., GOOG. For adjusted options, this root symbol may not match the current symbol.

expireDate	complex	Container for information on the expiration date
day	integer	Expiration day as a day of the month
month	integer	Expiration month as number
year	integer	Four-digit expiration year
expiryType	string	Expiry type of the option. Possible values are: QUARTERLY, MONTHLY, or WEEKLY.
product	complex	A container for information on the product specified in this option
exchangeCode	string	The primary exchange where the option is traded
symbol	string	The market symbol for the option, formatted as the stock symbol, expiration date, price, "Put" or "Call", and expiry type - "w" (weekly), "q" (quarterly), or neither (monthly). For example, "GOOG Jan 07 '11 \$660 Call w".
typeCode	string	Value is always OPTN. Supported for legacy purposes.
strikePrice	double	The agreed strike price for the option, as stated in the contract
callCount (or putCount)	integer	The number of call or put objects in this response
symbol	string	The market symbol for the instrument, e.g., GOOG

Sample Request

```
GET
https://etws.etrade.com/market/rest/optionchains?expirationMonth=04&expirationYear=2011&chainType=PUT&skipAdjusted=true&underlier=GOOG
```

Sample response - XML

```
<OptionChainResponse>
  <optionPairCount>2</optionPairCount>
  <optionPairs>
    <call>
      <rootSymbol>GOOG</rootSymbol>
      <expireDate>
        <day>7</day>
        <month>1</month>
        <year>2011</year>
        <expiryType>WEEKLY</expiryType>
      </expireDate>
      <product>
        <exchangeCode>CINC</exchangeCode>
        <symbol>GOOG Jan 07 '11 $540 Call w</symbol>
        <typeCode>OPTN</typeCode>
      </product>
      <strikePrice>540.000000</strikePrice>
    </call>
    <callCount>1</callCount>
    <pairType>CALLPUT</pairType>
    <put>
      <rootSymbol>GOOG</rootSymbol>
      <expireDate>
```

```

    <day>7</day>
    <month>1</month>
    <year>2011</year>
    <expiryType>WEEKLY</expiryType>
  </expireDate>
  <product>
    <exchangeCode>CINC</exchangeCode>
    <symbol>GOOG Jan 07 '11 $540 Put w</symbol>
    <typeCode>OPTN</typeCode>
  </product>
  <strikePrice>540.000000</strikePrice>
</put>
<putCount>1</putCount>
</optionPairs>
<optionPairs>
  <call>
    <rootSymbol>GOOG</rootSymbol>
    <expireDate>
      <day>7</day>
      <month>1</month>
      <year>2011</year>
      <expiryType>WEEKLY</expiryType>
    </expireDate>
    <product>
      <exchangeCode>CINC</exchangeCode>
      <symbol>GOOG Jan 07 '11 $660 Call w</symbol>
      <typeCode>OPTN</typeCode>
    </product>
    <strikePrice>660.000000</strikePrice>
  </call>
  <callCount>1</callCount>
  <pairType>CALLPUT</pairType>
  <put>
    <rootSymbol>GOOG</rootSymbol>
    <expireDate>
      <day>7</day>
      <month>1</month>
      <year>2011</year>
      <expiryType>WEEKLY</expiryType>
    </expireDate>
    <product>
      <exchangeCode>CINC</exchangeCode>
      <symbol>GOOG Jan 07 '11 $660 Put w</symbol>
      <typeCode>OPTN</typeCode>
    </product>
    <strikePrice>660.000000</strikePrice>
  </put>
  <putCount>1</putCount>
</optionPairs>
  <symbol>goog</symbol>
</OptionChainResponse>

```

Sample response - JSON

```
{
  "OptionChainResponse": {
    "optionPairCount": "2",
    "optionPairs": [
      {
        "call": {
          "rootSymbol": "GOOG",
          "expireDate": {
            "day": "7",
            "month": "1",
            "year": "2011",
            "expiryType": "WEEKLY"
          },
          "product": {
            "exchangeCode": "CINC",
            "symbol": "GOOG Jan 07 '11 $540 Call w",
            "typeCode": "OPTN"
          },
          "strikePrice": "540.000000"
        },
        "callCount": "1",
        "pairType": "CALLPUT",
        "put": {
          "rootSymbol": "GOOG",
          "expireDate": {
            "day": "7",
            "month": "1",
            "year": "2011",
            "expiryType": "WEEKLY"
          },
          "product": {
            "exchangeCode": "CINC",
            "symbol": "GOOG Jan 07 '11 $540 Put w",
            "typeCode": "OPTN"
          },
          "strikePrice": "540.000000"
        },
        "putCount": "1"
      },
      {
        "call": {
          "rootSymbol": "GOOG",
          "expireDate": {
            "day": "7",
            "month": "1",
            "year": "2011",
            "expiryType": "WEEKLY"
          },
          "product": {
            "exchangeCode": "CINC",
            "symbol": "GOOG Jan 07 '11 $660 Call w",
            "typeCode": "OPTN"
          },
          "strikePrice": "660.000000"
        },
        "callCount": "1",
        "pairType": "CALLPUT",
        "put": {
          "rootSymbol": "GOOG",
          "expireDate": {
            "day": "7",
            "month": "1",
            "year": "2011",
            "expiryType": "WEEKLY"
          },
          "product": {
            "exchangeCode": "CINC",
            "symbol": "GOOG Jan 07 '11 $660 Put w",
            "typeCode": "OPTN"
          },
          "strikePrice": "660.000000"
        },
        "putCount": "1"
      }
    ]
  }
}
```

```

        "strikePrice": "660.000000"
    },
    "callCount": "1",
    "pairType": "CALLPUT",
    "put": {
        "rootSymbol": "GOOG",
        "expireDate": {
            "day": "7",
            "month": "1",
            "year": "2011",
            "expiryType": "WEEKLY"
        },
        "product": {
            "exchangeCode": "CINC",
            "symbol": "GOOG Jan 07 '11 $660 Put w",
            "typeCode": "OPTN"
        },
        "strikePrice": "660.000000"
    },
    "putCount": "1"
}
1,
"symbol": "goog"
}
}

```

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Options display	Create a table of options for a desired security by requesting option data (with Get Option Chains), grouping it by dates (from Get Option Expire Dates), sorting each group, and displaying the result. May enhance display by using streaming quote updates for the symbol.	Get Option Chains, Get Option Expire Dates, Get Quote, Streaming API

Related APIs

Get Option Expire Dates, Get Quote

Sandbox Samples

The following shows a typical request and response in the sandbox environment.

Request

```

GET
https://etwssandbox.etrade.com/market/sandbox/rest/optionchains?chainType=PUT&expirati
onMonth=1&expirationYear=2013&underlier=GOOG

```

Response

```
<OptionChainResponse>
  <optionPairCount>2</optionPairCount>
  <optionPairs>
    <callCount>0</callCount>
    <pairType>PUTONLY</pairType>
    <put>
      <rootSymbol>GOU</rootSymbol>
      <expireDate>
        <day>20</day>
        <month>3</month>
        <year>2010</year>
        <expiryType>MONTHLY</expiryType>
      </expireDate>
      <product>
        <exchangeCode>CINC</exchangeCode>
        <symbol>GOU Mar 20 &apos;10 $240 Put</symbol>
        <typeCode>OPTN</typeCode>
      </product>
      <strikePrice>240.000000</strikePrice>
    </put>
    <putCount>1</putCount>
  </optionPairs>
  <optionPairs>
    <callCount>0</callCount>
    <pairType>PUTONLY</pairType>
    <put>
      <rootSymbol>GOU</rootSymbol>
      <expireDate>
        <day>20</day>
        <month>3</month>
        <year>2010</year>
        <expiryType>MONTHLY</expiryType>
      </expireDate>
      <product>
        <exchangeCode>CINC</exchangeCode>
        <symbol>GOU Mar 20 &apos;10 $250 Put</symbol>
        <typeCode>OPTN</typeCode>
      </product>
      <strikePrice>250.000000</strikePrice>
    </put>
    <putCount>1</putCount>
  </optionPairs>
  <symbol>GOOG</symbol>
</OptionChainResponse>
```


Get Option Expire Dates

Returns a list of expiration dates for options that have the specified underlier.

Description

Returns a list of dates suitable for structuring an option table display. The dates are used to group option data (returned by the `optionchains` method) for a specified underlier, creating a table display.

URL

<https://etws.etrade.com/market/rest/optionexpiredate>

HTTP Method: GET

Request Parameters

Parameter	Type	Required?	Description
underlier	string	required	The symbol for the underlying instrument for this option

Response Properties

Property	Type	Description
ExpirationDate	complex	Container for information on a single expiration date. There will be one of these for each date in the response.
day	integer	The day (1-31) the option will expire
month	integer	The month (1-12) the option will expire
year	integer	The 4-digit year the option will expire
expiryType	string	Expiry type of the option. Possible values are: QUARTERLY, MONTHLY, or WEEKLY.

Sample Request

```
GET https://etws.etrade.com/market/rest/optionexpiredate?underlier=ABC
```

Sample response - XML

```
<OptionExpireDateGetResponse>
  <ExpirationDate>
    <day>22</day>
    <month>1</month>
    <year>2011</year>
    <expiryType>MONTHLY</expiryType>
  </ExpirationDate>
  <ExpirationDate>
    <day>19</day>
    <month>2</month>
```

```

    <year>2011</year>
    <expiryType>MONTHLY</expiryType>
  </ExpirationDate>
</OptionExpireDateGetResponse>

```

Sample response - JSON

```

{
  "OptionExpireDateGetResponse": {
    "ExpirationDate": [
      {
        "day": "22",
        "month": "1",
        "year": "2011",
        "expiryType": "MONTHLY"
      },
      {
        "day": "19",
        "month": "2",
        "year": "2011",
        "expiryType": "MONTHLY"
      }
    ]
  }
}

```

Notes

- To create an option chain display, group the option chain data by dates, which are supplied by the Get Option Expire Dates API.

Sample use cases

Purpose	Workflow	Related APIs
Options display	Create a table of options for a desired security by requesting option data (with Get Option Chains), grouping it by dates (from Get Option Expire Dates), sorting each group, and displaying the result. May enhance display by using streaming quote updates for the symbol.	Get Option Chains, Get Option Expire Dates, Get Quote, Streaming API

Related APIs

Get Option Chains, Get Quote

Sandbox Samples

The following shows a typical request and response in the sandbox environment.

Request

```
GET
https://etwssandbox.etrade.com/market/sandbox/rest/optionexpireddate.json?underlier=IBM
```

Response

```
{
  "optionExpireDateGetResponse": {
    "expireDates": [
      {
        "day": 22,
        "month": 1,
        "year": 2011,
        "expiryType": "MONTHLY"
      },
      {
        "day": 19,
        "month": 2,
        "year": 2011,
        "expiryType": "MONTHLY"
      },
      {
        "day": 19,
        "month": 3,
        "year": 2011,
        "expiryType": "MONTHLY"
      },
      {
        "day": 18,
        "month": 6,
        "year": 2011,
        "expiryType": "MONTHLY"
      },
      {
        "day": 21,
        "month": 1,
        "year": 2012,
        "expiryType": "MONTHLY"
      },
      {
        "day": 19,
        "month": 1,
        "year": 2013,
        "expiryType": "MONTHLY"
      }
    ]
  }
}
```

Orders

List Orders

Returns a list of open orders or orders updated during the current day.

Description

This API returns a detailed list of current orders. It includes all orders that are currently open or were created, modified, executed, or cancelled during the current day (Eastern time). For option orders that have multiple legs, all legs are returned.

This API may return types of orders which cannot be created using version v0 of the E*TRADE Developer Platform, including multi-leg orders and conditional orders. Such orders were usually created via the E*TRADE website, mobile client, or other channel. See the tables below for details on the data structures of these orders.

This documentation assumes some familiarity with market terminology. For definitions of market terms, refer to the [E*TRADE Financial Help Center](#).

Since an account may have too many orders to display on one screen, the API provides for paging through them in small groups by specifying how many to return at a time (the `count`) and the starting point for each group (the `marker`).

URL

<https://etws.etrade.com/order/rest/orderlist/{accountId}>

HTTP Method: GET

Request Parameters

Parameter	Type	Required?	Description
accountId	path	required	Numeric account ID
marker	string	optional	Specifies the desired starting point of the set of items to return. Used for paging as described in the Notes below.
count	integer	optional	The number of orders to return in the response. The default is 25. Used for paging, as described in the Notes below.

Response Properties

The three tables below show a normal order, a group order, and the order element that is contained in both of those.

Normal order

Property	Type	Description
count	integer	The number of orders included in this response. Always equal to or less than the <i>count</i> specified in the request (which is 25 by default).
marker	string	Placeholder for paging through additional orders in the account, as described in the Notes below. An empty or absent marker indicates that there are no more orders beyond this set.
orderDetails	complex	Container for one or more sets of order details
OrderDetails	complex	Container for one set of order details
order	complex	Container for information on an order (<i>shown as a separate table below</i>)

Group order

Property	Type	Description
count	integer	The number of orders included in this response. Always equal to or less than the <i>count</i> specified in the request (which is 25 by default).
marker	string	Placeholder for paging through additional orders in the account, as described in the Notes below. An empty or absent marker indicates that there are no more orders beyond this set.
groupOrder	complex	Container for the items that follow
groupOrderId	long	The ID number for this group order
groupOrderType	string	The type of group order. Possible values are: <ul style="list-style-type: none">• CONTINGENT• ONE_CANCELS_ALL• ONE_TRIGGERS_ALL• ONE_TRIGGERS_OCO• BRACKETED For more information on these values, refer to the E*TRADE online help on conditional orders .
cumulativeEstimatedCommission	double	The cumulative estimated commission for this group order
cumulativeEstimatedFees	double	The cumulative estimated fee for this group order
order	complex	Container for information on an order (<i>shown as a separate table below</i>)

Order element

Property	Type	Description
order	complex	Container for information on an order
orderId	long	ID number assigned to this order
orderPlacedTime	long	The time the order was placed, in epoch time

orderExecutedTime	long	The time the order was placed, in epoch time
orderValue	double	Total cost or proceeds, including commission
orderStatus	string	The current status of the order. Possible values are: <ul style="list-style-type: none"> • OPEN • EXECUTED • CANCELLED • CANCEL_REQUESTED • EXPIRED • REJECTED
orderType	string	The order type. Possible values are: <ul style="list-style-type: none"> • EQ • OPTN • ADVANCE_EQ • ADVANCE_OPTN • SPREADS • BUY_WRITES • CONTINGENT • ONE_CANCELS_ALL • ONE_TRIGGERS_ALL • ONE_TRIGGERS_OCO • BRACKETED For more information, refer to the E*TRADE online help on conditional orders .
orderTerm	string	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL • FILL_OR_KILL For definitions

priceType	string	<p>The type of pricing. Possible values are:</p> <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • TRAILING_STOP_CNST_BY_LOWER_TRIGGER • UPPER_TRIGGER_BY_TRAILING_STOP_CNST • TRAILING_STOP_PRCT_BY_LOWER_TRIGGER • UPPER_TRIGGER_BY_TRAILING_STOP_PRCT • TRAILING_STOP_CNST • TRAILING_STOP_PRCT • HIDDEN_STOP • HIDDEN_STOP_BY_LOWER_TRIGGER • UPPER_TRIGGER_BY_HIDDEN_STOP • NET_DEBIT • NET_CREDIT • NET_EVEN • MARKET_ON_OPEN • MARKET_ON_CLOSE • LIMIT_ON_OPEN • LIMIT_ON_CLOSE <p>For more information on these values, refer to the E*TRADE online help on conditional orders.</p>
limitPrice	double	The highest price at which to buy or the lowest price at which to sell if specified in a limit order
stopPrice	double	The designated boundary price for a stop order
stopLimitPrice	double	The designated boundary price for a stop-limit order
routingDestination	string	<p>The exchange where the user requests that the order be executed. Possible values are:</p> <ul style="list-style-type: none"> • AUTO (default) • ARCA • NSDQ • NYSE • AMEX • BOX • CBOE • ISE • NOM • PHX
replacedByOrderId	long	The order ID of the order that was replaced by this order due to a change order request
allOrNone	boolean	If TRUE, the transactions specified in the order must be executed all at once, or not at all. Default is FALSE.
bracketLimitPrice	double	The user's lower trigger price on a buy and upper trigger price on a sale

initialStopPrice	double	The initial stop price that was set when a trailing stop order was submitted
trailPrice	double	The current trailing value. For Trailing Stop Dollar orders, this is a fixed dollar amount. For Trailing Stop Percentage orders, this is the price reflected by the percentage selected.
triggerPrice	double	The price that an advanced order will trigger. For example, if it's a \$1 buy trailing stop, then the trigger price will be \$1 above the last price.
conditionPrice	double	For a conditional order, the price the condition is being compared against
conditionSymbol	symbol	For a conditional order, the symbol that the condition is competing against
conditionType	string	The type of comparison to be used in a conditional order. Possible values are: <ul style="list-style-type: none"> • CONTINGENT_GTE • CONTINGENT_LTE For more information on these values, refer to the E*TRADE online help on conditional orders .
conditionFollowPrice	string	In a conditional order, the type of price being followed. Possible values are: ASK, BID, LAST.
legDetails	complex	Container for information on one leg of an order. There is at least one of these <code>legDetails</code> elements in every Order record, and more than one in a multi-leg order.
legNumber	long	The number associated with this leg of the order. For example, buy-write orders have two legs.
symbolDescription	string	Text description of the security
orderAction	string	The action that the broker is requested to perform. Possible values are: <ul style="list-style-type: none"> • BUY • SELL • BUY_TO_COVER • SELL_SHORT • BUY_OPEN • SELL_OPEN • BUY_CLOSE • SELL_CLOSE
orderedQuantity	double	The number of shares ordered
filledQuantity	double	The number of shares actually filled in the order
executedPrice	double	The price the stock was actually purchased for
estimatedCommission	double	The cost billed to the user to perform the requested action
estimatedFees	double	The total amount of fees billed to the user to fill the order
reserveQuantity	double	The number of shares to be publicly displayed if this is a reserve order

symbolInfo	complex	Container for symbol information for this leg. An option order will contain all of the following items; an equity order will only contain <code>symbol</code> .
symbol	string	The market symbol for the share being bought or sold
callPut	string	Specifies whether options are being bought (call) or sold (put). Possible values are: CALL, PUT.
expYear	integer	The 4-digit year the option will expire
expMonth	integer	The month (1-12) the option will expire
expDay	integer	The day (1-31) the option will expire
strikePrice	double	The strike price for the option specified in the order

Sample Request

```
GET https://etws.etrade.com/orders/rest/orderlist/83405188?count=10
```

Sample response - XML

```
<GetOrderListResponse>
  <orderListResponse>
    <count>2</count>
    <marker>1270023312|296</marker>
    <orderDetails>
      <OrderDetails>
        <order>
          <orderId>320</orderId>
          <orderPlacedTime>1270036641807</orderPlacedTime>
          <orderValue>1068.77</orderValue>
          <orderStatus>OPEN</orderStatus>
          <orderType>BUY_WRITES</orderType>
          <orderTerm>GOOD_FOR_DAY</orderTerm>
          <priceType>MARKET</priceType>
          <limitPrice>0</limitPrice>
          <stopPrice>0</stopPrice>
          <legDetails>
            <LegDetails>
              <legNumber>1</legNumber>
              <symbolInfo>
                <symbol>DELL</symbol>
              </symbolInfo>
              <symbolDescription>DELL INC COM</symbolDescription>
              <orderAction>BUY</orderAction>
              <orderedQuantity>100</orderedQuantity>
              <filledQuantity>0</filledQuantity>
              <executedPrice>0</executedPrice>
              <estimatedCommission>7.99</estimatedCommission>
              <estimatedFees>0</estimatedFees>
            </LegDetails>
          </legDetails>
        </order>
      </OrderDetails>
    </orderDetails>
  </orderListResponse>
</GetOrderListResponse>
```

```

        <legNumber>2</legNumber>
        <symbolInfo>
            <symbol>DLY</symbol>
            <callPut>CALL</callPut>
            <expYear>2010</expYear>
            <expMonth>4</expMonth>
            <expDay>17</expDay>
            <strikePrice>10</strikePrice>
        </symbolInfo>
        <symbolDescription>DELL APR 10 Call</symbolDescription>
        <orderAction>SELL_OPEN</orderAction>
        <orderedQuantity>1</orderedQuantity>
        <filledQuantity>0</filledQuantity>
        <executedPrice>0</executedPrice>
        <estimatedCommission>8.74</estimatedCommission>
        <estimatedFees>0</estimatedFees>
    </LegDetails>
</legDetails>
    <allOrNone>false</allOrNone>
</order>
</OrderDetails>
<OrderDetails>
    <order>
        <orderId>317</orderId>
        <orderPlacedTime>1270031791517</orderPlacedTime>
        <orderExecutedTime>1270031819000</orderExecutedTime>
        <orderValue>242.61</orderValue>
        <orderStatus>EXECUTED</orderStatus>
        <orderType>EQ</orderType>
        <orderTerm>GOOD_FOR_DAY</orderTerm>
        <priceType>MARKET</priceType>
        <limitPrice>0</limitPrice>
        <stopPrice>0</stopPrice>
        <legDetails>
            <LegDetails>
                <legNumber>1</legNumber>
                <symbolInfo>
                    <symbol>MSFT</symbol>
                </symbolInfo>
                <symbolDescription>MICROSOFT CORP COM</symbolDescription>
                <orderAction>SELL_SHORT</orderAction>
                <orderedQuantity>10</orderedQuantity>
                <filledQuantity>10</filledQuantity>
                <executedPrice>1</executedPrice>
                <estimatedCommission>7.99</estimatedCommission>
                <estimatedFees>0</estimatedFees>
            </LegDetails>
        </legDetails>
        <allOrNone>false</allOrNone>
    </order>
</OrderDetails>
</orderDetails>
</orderListResponse>
</GetOrderListResponse>

```

Sample response – JSON

```
{
  "GetOrderListResponse": {
    "orderListResponse": {
      "count": "2",
      "marker": "1270023312|296",
      "orderDetails": {
        "OrderDetails": [
          {
            "order": {
              "orderId": "320",
              "orderPlacedTime": "1270036641807",
              "orderValue": "1068.77",
              "orderStatus": "OPEN",
              "orderType": "BUY_WRITES",
              "orderTerm": "GOOD_FOR_DAY",
              "priceType": "MARKET",
              "limitPrice": "0",
              "stopPrice": "0",
              "legDetails": {
                "LegDetails": [
                  {
                    "legNumber": "1",
                    "symbolInfo": { "symbol": "DELL" },
                    "symbolDescription": "DELL INC COM",
                    "orderAction": "BUY",
                    "orderedQuantity": "100",
                    "filledQuantity": "0",
                    "executedPrice": "0",
                    "estimatedCommission": "7.99",
                    "estimatedFees": "0"
                  },
                  {
                    "legNumber": "2",
                    "symbolInfo": {
                      "symbol": "DLY",
                      "callPut": "CALL",
                      "expYear": "2010",
                      "expMonth": "4",
                      "expDay": "17",
                      "strikePrice": "10"
                    },
                    "symbolDescription": "DELL APR 10 Call",
                    "orderAction": "SELL_OPEN",
                    "orderedQuantity": "1",
                    "filledQuantity": "0",
                    "executedPrice": "0",
                    "estimatedCommission": "8.74",
                    "estimatedFees": "0"
                  }
                ]
              }
            },
            "allOrNone": "FALSE"
          }
        ]
      }
    }
  }
}
```

```

    }
  },
  {
    "order": {
      "orderId": "317",
      "orderPlacedTime": "1270031791517",
      "orderExecutedTime": "1270031819000",
      "orderValue": "242.61",
      "orderStatus": "EXECUTED",
      "orderType": "EQ",
      "orderTerm": "GOOD_FOR_DAY",
      "priceType": "MARKET",
      "limitPrice": "0",
      "stopPrice": "0",
      "legDetails": {
        "LegDetails": {
          "legNumber": "1",
          "symbolInfo": { "symbol": "MSFT" },
          "symbolDescription": "MICROSOFT CORP COM",
          "orderAction": "SELL_SHORT",
          "orderedQuantity": "10",
          "filledQuantity": "10",
          "executedPrice": "1",
          "estimatedCommission": "7.99",
          "estimatedFees": "0"
        }
      },
      "allOrNone": "FALSE"
    }
  }
]
}
}
}
}
}

```

Notes

- For complex options, only the first leg of the option shows the actual order status. Subsequent legs show the status as OPEN.
- For equity and option orders, types STOP and STOP LIMIT represent the ["Stop On Quote"](#) and ["Stop Limit on Quote"](#) price types offered by E*TRADE, respectively.
- For equity orders, use limit price for type LIMIT, stop price for type STOP, and both prices for type STOP_LIMIT. For option orders, use limit limit price for type LIMIT, stop price for type STOP, and stop-limit price for type STOP_LIMIT.
- To page through a large number of items, use the `count` property to specify how many items to return in a group (the default is 25), and the `marker` property to specify the starting point (the default is the newest). For instance, a request with no `count` and no `marker` retrieves the newest 25 items for the account. Each response includes a marker that points to the beginning of the next group. To page through all the items,

repeat the request with the marker from each previous response until you receive a response with an empty marker, indicating that there are no more items.

- The API does not explicitly provide for bi-directional paging. Your application can support paging backward and forward either by saving and re-using markers within the series (that is, to re-issue the requests for earlier pages in the series), or saving and re-displaying the items that come in.

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Current orders	Load all orders into array; display as a table with a droplist that filters by order status. For orders with status OPEN, show current quote info, and display an option to cancel the order.	Get Quote, Cancel Order
Display order status	As an advanced feature, register for push notifications on the account(s) being displayed. Have a listener process (e.g., Ajax) ready to display order status updates when they come in.	<i>See Notifications and Streaming API documentation for details.</i>

Related APIs

Get Quote, Cancel Order

Sandbox Samples

The following is an example of a request and response in the sandbox environment.

Request

```
GET https://etwssandbox.etrade.com/order/sandbox/rest/orderlist/83550325
```

Response

```
<GetOrderListResponse>
  <orderListResponse>
    <count>11</count>
    <marker></marker>
    <orderDetails>
      <OrderDetails>
        <order>
          <orderId>208</orderId>
          <orderPlacedTime>1267660551466</orderPlacedTime>
          <orderExecutedTime>1267660558000</orderExecutedTime>
          <orderValue>0</orderValue>
          <orderStatus>EXECUTED</orderStatus>
          <orderType>EQ</orderType>
        </order>
      </OrderDetails>
    </orderDetails>
  </orderListResponse>
</GetOrderListResponse>
```

```

<orderTerm>GOOD_FOR_DAY</orderTerm>
<priceType>MARKET</priceType>
<limitPrice>0</limitPrice>
<stopPrice>0</stopPrice>
<legDetails>
  <LegDetails>
    <legNumber>1</legNumber>
    <symbolInfo>
      <symbol>CSCO</symbol>
    </symbolInfo>
    <symbolDescription>CISCO SYS INC COM</symbolDescription>
    <orderAction>SELL</orderAction>
    <orderedQuantity>1</orderedQuantity>
    <filledQuantity>1</filledQuantity>
    <executedPrice>24.84</executedPrice>
    <estimatedCommission>5</estimatedCommission>
    <estimatedFees>0</estimatedFees>
  </LegDetails>
</legDetails>
<allOrNone>false</allOrNone>
</order>
</OrderDetails>
<OrderDetails>
  <order>
    <orderId>205</orderId>
    <orderPlacedTime>1267099216308</orderPlacedTime>
    <orderValue>4999994.21</orderValue>
    <orderStatus>OPEN</orderStatus>
    <orderType>OPTN</orderType>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <priceType>LIMIT</priceType>
    <limitPrice>50000</limitPrice>
    <stopPrice>0</stopPrice>
    <legDetails>
      <LegDetails>
        <legNumber>1</legNumber>
        <symbolInfo>
          <symbol>MSQ</symbol>
          <callPut>CALL</callPut>
          <expYear>2010</expYear>
          <expMonth>4</expMonth>
          <expDay>17</expDay>
          <strikePrice>26</strikePrice>
        </symbolInfo>
        <symbolDescription>MSFT APR 26 Call</symbolDescription>
        <orderAction>SELL_OPEN</orderAction>
        <orderedQuantity>1</orderedQuantity>
        <filledQuantity>0</filledQuantity>
        <executedPrice>0</executedPrice>
        <estimatedCommission>5.75</estimatedCommission>
        <estimatedFees>0</estimatedFees>
      </LegDetails>
    </legDetails>
    <allOrNone>false</allOrNone>
  </order>
</OrderDetails>

```

```

    </order>
  </OrderDetails>
  <OrderDetails>
    <order>
      <orderId>200</orderId>
      <orderPlacedTime>1267095453128</orderPlacedTime>
      <orderValue>210.79</orderValue>
      <orderStatus>CANCEL_REQUESTED</orderStatus>
      <orderType>BUY_WRITES</orderType>
      <orderTerm>GOOD_FOR_DAY</orderTerm>
      <priceType>NET_DEBIT</priceType>
      <limitPrice>2</limitPrice>
      <stopPrice>0</stopPrice>
      <legDetails>
        <LegDetails>
          <legNumber>1</legNumber>
          <symbolInfo>
            <symbol>IBM</symbol>
          </symbolInfo>
          <symbolDescription>INTERNATIONAL BUSINESS MACHS COM</symbolDescription>
          <orderAction>BUY</orderAction>
          <orderedQuantity>100</orderedQuantity>
          <filledQuantity>0</filledQuantity>
          <executedPrice>0</executedPrice>
          <estimatedCommission>5</estimatedCommission>
          <estimatedFees>0</estimatedFees>
        </LegDetails>
        <LegDetails>
          <legNumber>2</legNumber>
          <symbolInfo>
            <symbol>IBM</symbol>
            <callPut>CALL</callPut>
            <expYear>2010</expYear>
            <expMonth>4</expMonth>
            <expDay>17</expDay>
            <strikePrice>115</strikePrice>
          </symbolInfo>
          <symbolDescription>IBM APR 115 Call</symbolDescription>
          <orderAction>SELL_OPEN</orderAction>
          <orderedQuantity>1</orderedQuantity>
          <filledQuantity>0</filledQuantity>
          <executedPrice>0</executedPrice>
          <estimatedCommission>5.75</estimatedCommission>
          <estimatedFees>0</estimatedFees>
        </LegDetails>
      </legDetails>
      <allOrNone>false</allOrNone>
    </order>
  </OrderDetails>
  <OrderDetails>
    <order>
      <orderId>192</orderId>
      <orderPlacedTime>1267089996809</orderPlacedTime>
      <orderValue>10.79</orderValue>

```

```

<orderStatus>OPEN</orderStatus>
<orderType>OPTN</orderType>
<orderTerm>GOOD_FOR_DAY</orderTerm>
<priceType>LIMIT</priceType>
<limitPrice>0.05</limitPrice>
<stopPrice>0</stopPrice>
<legDetails>
  <LegDetails>
    <legNumber>1</legNumber>
    <symbolInfo>
      <symbol>QQQ</symbol>
      <callPut>CALL</callPut>
      <expYear>2010</expYear>
      <expMonth>3</expMonth>
      <expDay>20</expDay>
      <strikePrice>43</strikePrice>
    </symbolInfo>
    <symbolDescription>QQQQ MAR 43 Call</symbolDescription>
    <orderAction>BUY_OPEN</orderAction>
    <orderedQuantity>1</orderedQuantity>
    <filledQuantity>0</filledQuantity>
    <executedPrice>0</executedPrice>
    <estimatedCommission>5.75</estimatedCommission>
    <estimatedFees>0</estimatedFees>
  </LegDetails>
</legDetails>
<allOrNone>false</allOrNone>
</order>
</OrderDetails>
<OrderDetails>
  <order>
    <orderId>191</orderId>
    <orderPlacedTime>1267089623587</orderPlacedTime>
    <orderValue>15</orderValue>
    <orderStatus>OPEN</orderStatus>
    <orderType>EQ</orderType>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <priceType>LIMIT</priceType>
    <limitPrice>1</limitPrice>
    <stopPrice>0</stopPrice>
    <legDetails>
      <LegDetails>
        <legNumber>1</legNumber>
        <symbolInfo>
          <symbol>MSFT</symbol>
        </symbolInfo>
        <symbolDescription>MICROSOFT CORP COM</symbolDescription>
        <orderAction>BUY</orderAction>
        <orderedQuantity>10</orderedQuantity>
        <filledQuantity>0</filledQuantity>
        <executedPrice>0</executedPrice>
        <estimatedCommission>5</estimatedCommission>
        <estimatedFees>0</estimatedFees>
      </LegDetails>
    </legDetails>
  </order>
</OrderDetails>

```



```

        </legDetails>
        <allOrNone>false</allOrNone>
    </order>
</OrderDetails>
<OrderDetails>
    <order>
        <orderId>190</orderId>
        <orderPlacedTime>1267089606681</orderPlacedTime>
        <orderValue>15</orderValue>
        <orderStatus>OPEN</orderStatus>
        <orderType>EQ</orderType>
        <orderTerm>GOOD_FOR_DAY</orderTerm>
        <priceType>LIMIT</priceType>
        <limitPrice>1</limitPrice>
        <stopPrice>0</stopPrice>
        <legDetails>
            <LegDetails>
                <legNumber>1</legNumber>
                <symbolInfo>
                    <symbol>MSFT</symbol>
                </symbolInfo>
                <symbolDescription>MICROSOFT CORP COM</symbolDescription>
                <orderAction>BUY</orderAction>
                <orderedQuantity>10</orderedQuantity>
                <filledQuantity>0</filledQuantity>
                <executedPrice>0</executedPrice>
                <estimatedCommission>5</estimatedCommission>
                <estimatedFees>0</estimatedFees>
            </LegDetails>
        </legDetails>
        <allOrNone>false</allOrNone>
    </order>
</OrderDetails>
<OrderDetails>
    <order>
        <orderId>189</orderId>
        <orderPlacedTime>1267089589812</orderPlacedTime>
        <orderValue>15</orderValue>
        <orderStatus>OPEN</orderStatus>
        <orderType>EQ</orderType>
        <orderTerm>GOOD_FOR_DAY</orderTerm>
        <priceType>LIMIT</priceType>
        <limitPrice>1</limitPrice>
        <stopPrice>0</stopPrice>
        <legDetails>
            <LegDetails>
                <legNumber>1</legNumber>
                <symbolInfo>
                    <symbol>IBM</symbol>
                </symbolInfo>
                <symbolDescription>INTERNATIONAL BUSINESS MACHS COM</symbolDescription>
                <orderAction>BUY</orderAction>
                <orderedQuantity>10</orderedQuantity>
                <filledQuantity>0</filledQuantity>
            </LegDetails>
        </legDetails>
    </order>
</OrderDetails>

```

```

        <executedPrice>0</executedPrice>
        <estimatedCommission>5</estimatedCommission>
        <estimatedFees>0</estimatedFees>
    </LegDetails>
</legDetails>
    <allOrNone>false</allOrNone>
</order>
</OrderDetails>
<OrderDetails>
    <order>
        <orderId>188</orderId>
        <orderPlacedTime>1267089356773</orderPlacedTime>
        <orderValue>132.59</orderValue>
        <orderStatus>OPEN</orderStatus>
        <orderType>EQ</orderType>
        <orderTerm>GOOD_FOR_DAY</orderTerm>
        <priceType>MARKET</priceType>
        <limitPrice>0</limitPrice>
        <stopPrice>0</stopPrice>
        <legDetails>
            <LegDetails>
                <legNumber>1</legNumber>
                <symbolInfo>
                    <symbol>IBM</symbol>
                </symbolInfo>
                <symbolDescription>INTERNATIONAL BUSINESS MACHS COM</symbolDescription>
                <orderAction>BUY</orderAction>
                <orderedQuantity>1</orderedQuantity>
                <filledQuantity>0</filledQuantity>
                <executedPrice>0</executedPrice>
                <estimatedCommission>5</estimatedCommission>
                <estimatedFees>0</estimatedFees>
            </LegDetails>
        </legDetails>
        <allOrNone>false</allOrNone>
    </order>
</OrderDetails>
<OrderDetails>
    <order>
        <orderId>187</orderId>
        <orderPlacedTime>1266904795303</orderPlacedTime>
        <orderValue>10.79</orderValue>
        <orderStatus>OPEN</orderStatus>
        <orderType>OPTN</orderType>
        <orderTerm>GOOD_FOR_DAY</orderTerm>
        <priceType>LIMIT</priceType>
        <limitPrice>0.05</limitPrice>
        <stopPrice>0</stopPrice>
        <legDetails>
            <LegDetails>
                <legNumber>1</legNumber>
                <symbolInfo>
                    <symbol>MSQ</symbol>
                    <callPut>CALL</callPut>

```

```

        <expYear>2010</expYear>
        <expMonth>4</expMonth>
        <expDay>17</expDay>
        <strikePrice>27</strikePrice>
    </symbolInfo>
    <symbolDescription>MSFT APR 27 Call</symbolDescription>
    <orderAction>BUY_OPEN</orderAction>
    <orderedQuantity>1</orderedQuantity>
    <filledQuantity>0</filledQuantity>
    <executedPrice>0</executedPrice>
    <estimatedCommission>5.75</estimatedCommission>
    <estimatedFees>0</estimatedFees>
</LegDetails>
</legDetails>
    <allOrNone>false</allOrNone>
</order>
</OrderDetails>
<OrderDetails>
    <order>
        <orderId>170</orderId>
        <orderPlacedTime>1266584434221</orderPlacedTime>
        <orderValue>391.944</orderValue>
        <orderStatus>OPEN</orderStatus>
        <orderType>OPTN</orderType>
        <orderTerm>GOOD_FOR_DAY</orderTerm>
        <priceType>LIMIT</priceType>
        <limitPrice>1</limitPrice>
        <stopPrice>0</stopPrice>
        <legDetails>
            <LegDetails>
                <legNumber>1</legNumber>
                <symbolInfo>
                    <symbol>IBM</symbol>
                    <callPut>CALL</callPut>
                    <expYear>2010</expYear>
                    <expMonth>2</expMonth>
                    <expDay>20</expDay>
                    <strikePrice>130</strikePrice>
                </symbolInfo>
                <symbolDescription>IBM FEB 130 Call</symbolDescription>
                <orderAction>SELL_OPEN</orderAction>
                <orderedQuantity>4</orderedQuantity>
                <filledQuantity>0</filledQuantity>
                <executedPrice>0</executedPrice>
                <estimatedCommission>8</estimatedCommission>
                <estimatedFees>0</estimatedFees>
            </LegDetails>
        </legDetails>
        <allOrNone>false</allOrNone>
    </order>
</OrderDetails>
<OrderDetails>
    <order>
        <orderId>166</orderId>

```

```

<orderPlacedTime>1266583451241</orderPlacedTime>
<orderValue>4020.28</orderValue>
<orderStatus>OPEN</orderStatus>
<orderType>OPTN</orderType>
<orderTerm>GOOD_FOR_DAY</orderTerm>
<priceType>LIMIT</priceType>
<limitPrice>2</limitPrice>
<stopPrice>0</stopPrice>
<legDetails>
  <LegDetails>
    <legNumber>1</legNumber>
    <symbolInfo>
      <symbol>IBM</symbol>
      <callPut>CALL</callPut>
      <expYear>2010</expYear>
      <expMonth>2</expMonth>
      <expDay>20</expDay>
      <strikePrice>130</strikePrice>
    </symbolInfo>
    <symbolDescription>IBM FEB 130 Call</symbolDescription>
    <orderAction>BUY_CLOSE</orderAction>
    <orderedQuantity>20</orderedQuantity>
    <filledQuantity>10</filledQuantity>
    <executedPrice>2</executedPrice>
    <estimatedCommission>20</estimatedCommission>
    <estimatedFees>0</estimatedFees>
  </LegDetails>
</legDetails>
<allOrNone>false</allOrNone>
</order>
</OrderDetails>
<OrderDetails>
  <order>
    <orderId>162</orderId>
    <orderPlacedTime>1266581179588</orderPlacedTime>
    <orderValue>4020.28</orderValue>
    <orderStatus>OPEN</orderStatus>
    <orderType>OPTN</orderType>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <priceType>LIMIT</priceType>
    <limitPrice>2</limitPrice>
    <stopPrice>0</stopPrice>
    <legDetails>
      <LegDetails>
        <legNumber>1</legNumber>
        <symbolInfo>
          <symbol>GOP</symbol>
          <callPut>CALL</callPut>
          <expYear>2010</expYear>
          <expMonth>3</expMonth>
          <expDay>20</expDay>
          <strikePrice>520</strikePrice>
        </symbolInfo>
        <symbolDescription>GOOG MAR 520 Call</symbolDescription>

```

```
        <orderAction>BUY_CLOSE</orderAction>
        <orderedQuantity>20</orderedQuantity>
        <filledQuantity>10</filledQuantity>
        <executedPrice>2</executedPrice>
        <estimatedCommission>20</estimatedCommission>
        <estimatedFees>0</estimatedFees>
    </LegDetails>
</legDetails>
    <allOrNone>false</allOrNone>
</order>
</OrderDetails>
</orderDetails>
</orderListResponse>
</GetOrderListResponse>
```

Preview Equity Order

Returns a preview of an equity order.

Description

This API accepts a simulated equity order as input and returns a preview of the order, with estimated costs, allowing the user to review the order before placing it.

The preview response includes a preview ID number which must be referenced when placing the actual order.

URL

<https://etws.etrade.com/order/rest/previewequityorder>

HTTP Method: POST

Since this is a POST request, the parameters are included in the request as XML or JSON.

Request Parameters

Parameter	Type	Required?	Description
accountId	integer	required	Numeric account ID
symbol	string	required	The market symbol for the security being bought or sold
orderAction	enum	required	The action that the broker is requested to perform. Possible values are: <ul style="list-style-type: none">• BUY• SELL• BUY_TO_COVER• SELL_SHORT
clientOrderId	string	required	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
priceType	enum	required	The type of pricing. Possible values are: <ul style="list-style-type: none">• MARKET• LIMIT• STOP• STOP_LIMIT• MARKET_ON_CLOSE If STOP, requires a <code>stopPrice</code> . If LIMIT, requires a <code>limitPrice</code> . If STOP_LIMIT equity order, requires both.
limitPrice	double	conditional	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Required if <code>priceType</code> is LIMIT or STOP_LIMIT.

stopPrice	double	conditional	The price at which to buy or sell if specified in a stop order. Required if <code>priceType</code> is STOP or STOP_LIMIT.
allOrNone	boolean	optional	If TRUE, the transactions specified in the order must be executed all at once, or not at all. Default is FALSE.
quantity	integer	required	The number of shares to buy or sell
reserveOrder	boolean	optional	If set to TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. Default is FALSE. If TRUE, must also specify the <code>reserveQuantity</code> .
reserveQuantity	integer	conditional	The number of shares to be publicly displayed if this is a reserve order. Required if <code>reserveOrder</code> is TRUE.
marketSession	enum	required	Session in which the equity order will be placed. Possible values are: REGULAR, EXTENDED.
orderTerm	enum	required	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
routingDestination	enum	optional	The exchange where the order should be executed. Users may want to specify this if they believe they can get a better order fill at a specific exchange, rather than relying on the automatic order routing system. Possible values are: <ul style="list-style-type: none"> • AUTO (default) • ARCA • NSDQ • NYSE

Response Properties

Property	Type	Description
accountId	integer	Numeric account ID
allOrNone	boolean	If TRUE, the transactions are to be executed all at once, or not at all.
estimatedCommission	double	The estimated commission
estimatedTotalAmount	double	The estimated total cost or proceeds, including commission
messageList	complex	Container for messages describing the result of the action
message	complex	Container for a result message
msgDesc	string	Text of the result message, indicating order status, success or failure, additional requirements that must be met before placing the order, etc. Applications typically display this message to the user, which may result in further user action.
msgCode	integer	Standard numeric code of the result message. Refer to the Error Messages documentation for examples. May optionally be displayed to the user, but is primarily intended for internal use.
previewTime	long	The time of this preview, in epoch time

previewId	long	Numeric preview ID
quantity	integer	The number of shares to buy or sell
reserveOrder	boolean	If TRUE, this is a reserve order - meaning that only a limited number of shares will be publicly displayed, instead of the entire order, to avoid influencing other traders.
reserveQuantity	integer	The number of shares to be publicly displayed if this is a reserve order
orderTerm	string	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
priceType	string	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE
limitPrice	double	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Returned if priceType is LIMIT.
stopPrice	double	The price at which a stock is to be bought or sold if specified in a stop order. Returned if priceType is STOP.
symbolDesc	string	Text description of the security
symbol	string	The market symbol for the stock being bought or sold
orderAction	string	The action the broker is requested to perform. Possible values are: <ul style="list-style-type: none"> • BUY • SELL • BUY_TO_COVER • SELL_SHORT
orderTime	long	The time the order was submitted
routingDestination	string	The exchange where the user requests that the order be executed. Possible values are: <ul style="list-style-type: none"> • AUTO • ARCA • NSDQ • NYSE

Sample Request

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/previewequityorder
```


Request Parameters - XML

```
<PreviewEquityOrder xmlns="http://order.etws.etrade.com">
  <EquityOrderRequest>
    <accountId>83405188</accountId>
    <limitPrice></limitPrice>
    <stopPrice>197</stopPrice>
    <allOrNone></allOrNone>
    <quantity>4</quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
    <symbol>IBM</symbol>
    <orderAction>BUY</orderAction>
    <priceType>STOP</priceType>
    <routingDestination></routingDestination>
    <marketSession>REGULAR</marketSession>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <clientOrderId>random123456</clientOrderId>
  </EquityOrderRequest>
</PreviewEquityOrder>
```

Request Parameters - JSON

```
{
  "PreviewEquityOrder": {
    "-xmlns": "http://order.etws.etrade.com",
    "EquityOrderRequest": {
      "accountId": "83405188",
      "stopPrice": "197",
      "quantity": "4",
      "symbol": "IBM",
      "orderAction": "BUY",
      "priceType": "STOP",
      "marketSession": "REGULAR",
      "orderTerm": "GOOD_FOR_DAY",
      "clientOrderId": "random123456"
    }
  }
}
```

Sample response - XML

```
<PreviewEquityOrderResponse>
  <equityOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>7.99</estimatedCommission>
    <estimatedTotalAmount>795.99</estimatedTotalAmount>
    <messageList>
      <message>
        <msgDesc>You have an existing open order for this security on the same
          side of the market. If you did not intend to place a second order
          for this security, please modify your order now.
        </msgDesc>
      </message>
    </messageList>
  </equityOrderResponse>
</PreviewEquityOrderResponse>
```

```

        </msgDesc>
        <msgCode>1042</msgCode>
    </message>
</messageList>
<previewTime>1269428745346</previewTime>
<previewId>449548380022</previewId>
<quantity>4</quantity>
<reserveOrder>false</reserveOrder>
<reserveQuantity>0</reserveQuantity>
<orderTerm>GOOD_FOR_DAY</orderTerm>
<limitPrice>0</limitPrice>
<stopPrice>197</stopPrice>
<symbolDesc>INTERNATIONAL BUSINESS MACHS COM</symbolDesc>
<symbol>IBM</symbol>
<orderAction>BUY</orderAction>
<priceType>STOP</priceType>
</equityOrderResponse>
</PreviewEquityOrderResponse>

```

Sample response – JSON

```

{
  "PreviewEquityOrderResponse": {
    "equityOrderResponse": {
      "accountId": "83405188",
      "allOrNone": "FALSE",
      "estimatedCommission": "7.99",
      "estimatedTotalAmount": "795.99",
      "messageList": {
        "message": {
          "msgDesc": "You have an existing open order for this security on the same
            side of the market. If you did not intend to place a second order
            for this security, please modify your order now.",
          "msgCode": "1042"
        }
      },
      "previewTime": "1269428745346",
      "previewId": "449548380022",
      "quantity": "4",
      "reserveOrder": "FALSE",
      "reserveQuantity": "0",
      "orderTerm": "GOOD_FOR_DAY",
      "limitPrice": "0",
      "stopPrice": "197",
      "symbolDesc": "INTERNATIONAL BUSINESS MACHS COM",
      "symbol": "IBM",
      "orderAction": "BUY",
      "priceType": "STOP"
    }
  }
}

```

Notes

- For equity and option orders, types STOP and STOP LIMIT represent the ["Stop On Quote"](#) and ["Stop Limit on Quote"](#) price types offered by E*TRADE, respectively.
- For equity orders, use limit price for priceType "LIMIT", stop price for priceType "STOP", and both prices for priceType "STOP_LIMIT". For option orders, use limit price for priceType "LIMIT", stop price for priceType "STOP", and stop-limit price for priceType "STOP_LIMIT".
- The `clientOrderID` element is used to ensure that a duplicate order is not inadvertently submitted. Within a given account, `clientOrderID` must be unique for every order (regardless of which APIs are used).

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Order preview	User interacts with screen to assemble an order by selecting an account to use and a product to trade. Display shows account ID, name, and balances, as well as product fundamentals, intraday, and current (streaming) price info. In a loop, whenever the user clicks to preview the estimated cost, call the preview API, reload the display from the response, and allow the user to continue editing, commit, or quit. The application should always validate all user inputs and also check the response for error messages.	Get Quote, List Accounts, Get Account Balance, Streaming API

Related APIs

Place Equity Order, Preview Equity Order Change, Preview Option Order, Get Quote, List Accounts, Get Account Balance

Sandbox Samples

The following is an example of a request and response in the sandbox environment. Note that the HTTP POST method is used.

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/previewequityorder
```

Request Parameters - XML

```
<PreviewEquityOrder xmlns="http://order.etws.etrade.com">
  <EquityOrderRequest>
    <symbol>CSCO</symbol>
```

```

    <orderAction>BUY</orderAction>
    <priceType>LIMIT</priceType>
    <routingDestination></routingDestination>
    <marketSession>REGULAR</marketSession>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <allOrNone></allOrNone>
    <quantity>4</quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity>0</reserveQuantity>
    <accountId>83550325</accountId>
    <clientOrderId>123123</clientOrderId>
    <limitPrice>10</limitPrice>
    <previewId></previewId>
  </EquityOrderRequest>
</PreviewEquityOrder>

```

Response

```

<PreviewEquityOrderResponse>
  <equityOrderResponse>
    <accountId>83310056</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>9.99</estimatedCommission>
    <estimatedTotalAmount>1909.99</estimatedTotalAmount>
    <previewTime>1266974074852</previewTime>
    <previewId>499500613087</previewId>
    <quantity>100</quantity>
    <reserveOrder>false</reserveOrder>
    <reserveQuantity>0</reserveQuantity>
    <orderTerm>GOOD_UNTIL_CANCEL</orderTerm>
    <limitPrice>0</limitPrice>
    <stopPrice>0</stopPrice>
    <symbolDesc>CISCO SYS INC COM</symbolDesc>
    <symbol>CSCO</symbol>
    <orderAction>BUY</orderAction>
    <priceType>LIMIT</priceType>
  </equityOrderResponse>
</PreviewEquityOrderResponse>

```

Place Equity Order

Submits an equity order.

Description

This API places an equity order. In a typical workflow, the order is first previewed with Preview Equity Order. Once it has been placed, the order can be viewed with the List Orders API.

Throughout this documentation, the terms "submit" and "place" refer to entering an order into the system to be executed. The actual execution of the order depends on a variety of circumstances. To receive timely notification of the order status, use the streaming order status API, detailed elsewhere in this documentation.

The related Preview Equity Order API returns a preview of an order that includes estimated order cost and a unique preview ID. It is possible to place an order without first previewing it, but a typical order workflow lets the user preview the order before submitting it. Be aware that, once an order has been previewed, placing the order requires that it include the preview ID and the same parameters that were used in the preview.

URL

<https://etws.etrade.com/order/rest/placeequityorder>

HTTP Method: POST

Since this is a POST request, the parameters are included in the request as XML or JSON.

Request Parameters

Parameter	Type	Required?	Description
accountId	integer	required	Numeric account ID
symbol	string	required	The market symbol for the security being bought or sold
orderAction	enum	required	The action that the broker is requested to perform. Possible values are: <ul style="list-style-type: none">• BUY• SELL• BUY_TO_COVER• SELL_SHORT
previewId	long	conditional	If the order was not previewed, this parameter should not be specified. If the order was previewed, this parameter must specify the numeric preview ID from the preview, and other parameters of this request must match the parameters of the preview.

clientOrderId	string	required	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
priceType	enum	required	The type of pricing specified in the equity order. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE If STOP, requires a <code>stopPrice</code> . If LIMIT, requires a <code>limitPrice</code> . If STOP_LIMIT, requires both. For more information on these values, refer to the E*TRADE online help on conditional orders .
limitPrice	double	conditional	The highest price at which to buy or the lowest price at which to sell. Required if <code>priceType</code> is LIMIT or STOP_LIMIT.
stopPrice	double	conditional	The price at which to buy or sell if specified in a stop order. Required if <code>priceType</code> is STOP or STOP_LIMIT.
allOrNone	boolean	optional	If TRUE, the transactions specified in the order must be executed all at once, or not at all. Default is FALSE.
quantity	integer	required	The number of shares to buy or sell
reserveOrder	boolean	optional	If set to TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. Default is FALSE. If TRUE, must also specify the <code>reserveQuantity</code> .
reserveQuantity	integer	conditional	The number of shares to be publicly displayed if this is a reserve order. Required if <code>reserveOrder</code> is TRUE.
marketSession	enum	required	Session in which the equity order will be placed. Possible values are: REGULAR, EXTENDED.
orderTerm	enum	required	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
routingDestination	enum	optional	The exchange where the order should be executed. Users may want to specify this if they believe they can get a better order fill at a specific exchange, rather than relying on the automatic order routing system. Possible values are: <ul style="list-style-type: none"> • AUTO (default) • ARCA • NSDQ • NYSE

Response Properties

Property	Type	Description
accountId	integer	Numeric account ID
allOrNone	boolean	If TRUE, the transactions specified in the order are to be executed all at once, or not at all.
estimatedCommission	double	The cost billed to the user to perform the requested action
estimatedTotalAmount	double	The cost or proceeds, including broker commission, resulting from the requested action.
messageList	complex	Container for messages describing the result of the action
message	complex	Container for a result message
msgDesc	string	Text of the result message, indicating order status, success or failure, additional requirements that must be met before placing the order, etc. Applications typically display this message to the user, which may result in further user action.
msgCode	integer	Standard numeric code of the result message. Refer to the Error Messages documentation for examples. May optionally be displayed to the user, but is primarily intended for internal use.
orderNum	integer	Numeric ID for this order
orderTime	long	The time the order was submitted, in epoch time
symbolDesc	string	Text description of the security
symbol	string	The market symbol for the underlier
quantity	integer	The number of shares to buy or sell
reserveOrder	boolean	If TRUE, this is a reserve order - meaning that only a limited number of shares will be publicly displayed, instead of the entire order, to avoid influencing other traders.
reserveQuantity	integer	The number of shares to be publicly displayed if this is a reserve order.
orderTerm	string	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
orderAction	string	The action that the broker is requested to perform. Possible values are: <ul style="list-style-type: none"> • BUY • SELL • BUY_TO_COVER • SELL_SHORT

priceType	string	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE
limitPrice	double	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Returned if priceType is LIMIT.
stopPrice	double	The price at which a stock is to be bought or sold if specified in a stop order. Returned if priceType is STOP.
routingDestination	string	The exchange where the order should be executed. Possible values are: <ul style="list-style-type: none"> • AUTO • ARCA • NSDQ • NYSE

Sample Request

Request URL

POST <https://etwssandbox.etrade.com/order/sandbox/rest/placeequityorder>

Request Parameters - XML

```
<PlaceEquityOrder xmlns="http://order.etws.etrade.com">
  <EquityOrderRequest>
    <accountId>83405188</accountId>
    <clientOrderId>45</clientOrderId>
    <limitPrice>3</limitPrice>
    <previewId></previewId>
    <stopPrice></stopPrice>
    <allOrNone></allOrNone>
    <quantity>4</quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
    <symbol>ETFC</symbol>
    <orderAction>BUY</orderAction>
    <priceType>LIMIT</priceType>
    <routingDestination></routingDestination>
    <marketSession>REGULAR</marketSession>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
  </EquityOrderRequest>
</PlaceEquityOrder>
```

Request Parameters - JSON

```
{
  "PlaceEquityOrder": {
    "-xmlns": "http://order.etws.etrade.com",
```



```

    "EquityOrderRequest": {
      "accountId": "83405188",
      "clientOrderId": "45",
      "limitPrice": "3",
      "quantity": "4",
      "symbol": "ETFC",
      "orderAction": "BUY",
      "priceType": "LIMIT",
      "marketSession": "REGULAR",
      "orderTerm": "GOOD_FOR_DAY"
    }
  }
}

```

Sample response - XML

```

<PlaceEquityOrderResponse>
  <equityOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>7.99</estimatedCommission>
    <estimatedTotalAmount>19.99</estimatedTotalAmount>
    <messageList>
      <message>
        <msgDesc>
          Your order was successfully entered during market hours.
        </msgDesc>
        <msgCode>1026</msgCode>
      </message>
    </messageList>
    <orderNum>13</orderNum>
    <orderTime>1269423948429</orderTime>
    <quantity>4</quantity>
    <reserveOrder>false</reserveOrder>
    <reserveQuantity>0</reserveQuantity>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <limitPrice>3</limitPrice>
    <stopPrice>0</stopPrice>
    <symbolDesc>E TRADE FINANCIAL CORP COM</symbolDesc>
    <symbol>ETFC</symbol>
    <orderAction>BUY</orderAction>
    <priceType>LIMIT</priceType>
  </equityOrderResponse>
</PlaceEquityOrderResponse>

```

Sample response – JSON

```

{
  "PlaceEquityOrderResponse": {
    "equityOrderResponse": {
      "accountId": "83405188",
      "allOrNone": "FALSE",

```

```

    "estimatedCommission": "7.99",
    "estimatedTotalAmount": "19.99",
    "messageList": {
      "message": {
        "msgDesc": "Your order was successfully entered during market hours",
        "msgCode": "1026"
      }
    },
    "orderNum": "13",
    "orderTime": "1269423948429",
    "quantity": "4",
    "reserveOrder": "FALSE",
    "reserveQuantity": "0",
    "orderTerm": "GOOD_FOR_DAY",
    "limitPrice": "3",
    "stopPrice": "0",
    "symbolDesc": "E TRADE FINANCIAL CORP COM",
    "symbol": "ETFC",
    "orderAction": "BUY",
    "priceType": "LIMIT"
  }
}

```

Notes

- For equity and option orders, types STOP and STOP LIMIT represent the ["Stop On Quote"](#) and ["Stop Limit on Quote"](#) price types offered by E*TRADE, respectively.
- For equity orders, use limit price for type LIMIT, stop price for type STOP, and both prices for type STOP_LIMIT. For option orders, use limit limit price for type LIMIT, stop price for type STOP, and stop-limit price for type STOP_LIMIT.
- The `clientOrderID` element is used to ensure that a duplicate order is not inadvertently submitted. Within a given account, `clientOrderID` must be unique for every order (regardless of which APIs are used).

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Place order	On the order screen, enable the menu option to place an order only when displaying the response from the preview API. In addition to the order preview, show account ID, name, and balances, as well as product fundamentals, intraday, and current (streaming) price info. Confirm that balance is sufficient to cover estimated cost before placing order with <code>placeequityorder</code> . Display response.	Get Quote, List Accounts, Get Account Balance, Streaming API

Display order status	When placing or changing an order, register for push notifications on the account. Have a listener process on each page appropriately - for instance, be prepared to display order status updates on the completed order display, order list display, or account display.	See <i>Notifications and Streaming API documentation</i> for details.
----------------------	---	---

Related APIs

Preview Equity Order, Place Equity Order Change, Place Option Order

Sandbox Samples

The following is an example of a request and response in the sandbox environment. Note that the HTTP POST method is used.

Request URL

POST <https://etwssandbox.etrade.com/order/sandbox/rest/placeequityorder.json>

Request Parameters - JSON

```
{
  "PlaceEquityOrder": {
    "-xmlns": "http://order.etws.etrade.com",
    "EquityOrderRequest": {
      "symbol": "CSCO",
      "orderAction": "BUY",
      "priceType": "LIMIT",
      "marketSession": "REGULAR",
      "orderTerm": "GOOD_FOR_DAY",
      "quantity": "4",
      "reserveQuantity": "0",
      "accountId": "83550325",
      "clientOrderId": "asfdsa12312",
      "limitPrice": "10",
      "stopPrice": "10"
    }
  }
}
```

Response

```
{
  "PlaceEquityOrderResponse": {
    "EquityOrderResponse": {
      "accountId": 83310056,
      "allOrNone": false,
      "estimatedCommission": 9.99,
      "estimatedTotalAmount": 1909.99,
      "messageList": {
```

```
    "msgDesc": "Your order was successfully entered during market hours.",
    "msgCode": 1026
  },
  "orderNum": 277,
  "orderTime": 1240982042179,
  "quantity": 100,
  "reserveOrder": false,
  "reserveQuantity": 0,
  "orderTerm": "GOOD_UNTIL_CANCEL",
  "limitPrice": 18,
  "stopPrice": 0,
  "symbolDesc": "CISCO SYS INC COM",
  "symbol": "CSCO",
  "orderAction": "BUY",
  "priceType": "LIMIT"
}
}
```

Preview Equity Order Change

Returns a preview of a change to an equity order.

Description

This API accepts a simulated equity order change as input and returns a preview of the modified order, with estimated costs, allowing the user to review the change before submitting it.

The preview response includes a preview ID number which must be referenced when submitting the actual change request.

URL

<https://etws.etrade.com/order/rest/previewchangeequityorder>

HTTP Method: POST

Since this is a POST request, the parameters are included in the request as XML or JSON.

Request Parameters

Parameter	Type	Required?	Description
accountId	integer	required	Numeric account ID
orderNum	integer	required	Order number, taken from a previous response, that identifies the order to be changed
priceType	enum	required	The type of pricing. Possible values are: <ul style="list-style-type: none">• MARKET• LIMIT• STOP• STOP_LIMIT• MARKET_ON_CLOSE If STOP, requires a <code>stopPrice</code> . If LIMIT, requires a <code>limitPrice</code> . If STOP_LIMIT, requires both.
limitPrice	double	conditional	The highest price at which to buy or the lowest price at which to sell. Required if <code>priceType</code> is LIMIT.
stopPrice	double	conditional	The price at which to buy or sell if specified in a stop order. Required if <code>priceType</code> is STOP.
allOrNone	boolean	optional	If TRUE, the transactions specified in the order must be executed all at once, or not at all. Default is FALSE.
quantity	integer	required	The number of shares to buy or sell
reserveOrder	boolean	optional	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. Default is FALSE. If TRUE, must also specify the <code>reserveQuantity</code> .
reserveQuantity	integer	conditional	The number of shares displayed for a reserve order. Required if <code>reserveOrder</code> is TRUE.

orderTerm	enum	required	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
clientOrderId	string	optional	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.

Response Properties

Property	Type	Description
accountId	integer	Numeric account ID
allOrNone	boolean	If TRUE, the transactions specified in the order must be executed all at once, or not at all.
estimatedCommission	double	The cost billed to the user to perform the requested action
estimatedTotalAmount	double	The cost or proceeds, including broker commission, resulting from the requested action
messageList	complex	Container for messages describing the result of the action
message	complex	Container for a result message
msgDesc	string	Text of the result message, indicating order status, success or failure, additional requirements that must be met before placing the order, etc. Applications typically display this message to the user, which may result in further user action.
msgCode	integer	Standard numeric code of the result message. Refer to the Error Messages documentation for examples. May optionally be displayed to the user, but is primarily intended for internal use.
previewTime	long	The time of this preview, in epoch time
previewId	long	Numeric preview ID
quantity	integer	The number of shares to buy or sell
reserveOrder	boolean	If TRUE, this is a reserve order - meaning that only a limited number of shares will be publicly displayed, instead of the entire order, to avoid influencing other traders.
reserveQuantity	integer	The number of shares to be publicly displayed if this is a reserve order
orderTerm	string	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)

priceType	string	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE
limitPrice	double	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Returned if priceType is LIMIT.
stopPrice	double	The price at which a stock is to be bought or sold if specified in a stop order. Returned if priceType is STOP.
symbolDesc	string	Text description of the security being bought or sold
symbol	string	The market symbol for the security being bought or sold
orderAction	string	The action that the broker is requested to perform. Possible values are: <ul style="list-style-type: none"> • BUY • SELL • BUY_OPEN • SELL_OPEN

Sample Request

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/changeequityorder
```

Request Parameters - XML

```
<previewChangeEquityOrder xmlns="http://order.etws.etrade.com">
  <changeEquityOrderRequest>
    <accountId>83405188</accountId>
    <orderNum>16</orderNum>
    <limitPrice></limitPrice>
    <stopPrice></stopPrice>
    <allOrNone></allOrNone>
    <quantity></quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
    <priceType></priceType>
    <orderTerm>GOOD_UNTIL_CANCEL</orderTerm>
  </changeEquityOrderRequest>
</previewChangeEquityOrder>
```

Request Parameters - JSON

```
{
  "previewChangeEquityOrder": {
    "-xmlns": "http://order.etws.etrade.com",
    "changeEquityOrderRequest": {
      "accountId": "83405188",
```

```

        "orderNum": "16",
        "orderTerm": "GOOD_UNTIL_CANCEL"
    }
}
}

```

Sample response - XML

```

<PreviewChangeEquityOrderResponse>
  <equityOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>7.99</estimatedCommission>
    <estimatedTotalAmount>19.99</estimatedTotalAmount>
    <messageList/>
    <orderNum>0</orderNum>
    <orderTime>1269430328285</orderTime>
    <previewTime>1269430496444</previewTime>
    <previewId>449548414022</previewId>
    <quantity>4</quantity>
    <reserveOrder>false</reserveOrder>
    <reserveQuantity>0</reserveQuantity>
    <orderTerm>GOOD_UNTIL_CANCEL</orderTerm>
    <limitPrice>3</limitPrice>
    <symbolDesc>E TRADE FINANCIAL CORP COM</symbolDesc>
    <symbol>ETFC</symbol>
    <orderAction>BUY</orderAction>
    <priceType>LIMIT</priceType>
  </equityOrderResponse>
</PreviewChangeEquityOrderResponse>

```

Sample response – JSON

```

{
  "PreviewChangeEquityOrderResponse": {
    "equityOrderResponse": {
      "accountId": "83405188",
      "allOrNone": "FALSE",
      "estimatedCommission": "7.99",
      "estimatedTotalAmount": "19.99",
      "orderNum": "0",
      "orderTime": "1269430328285",
      "previewTime": "1269430496444",
      "previewId": "449548414022",
      "quantity": "4",
      "reserveOrder": "FALSE",
      "reserveQuantity": "0",
      "orderTerm": "GOOD_UNTIL_CANCEL",
      "limitPrice": "3",
      "symbolDesc": "E TRADE FINANCIAL CORP COM",
      "symbol": "ETFC",
      "orderAction": "BUY",

```



```

    "priceType": "LIMIT"
  }
}
}

```

Notes

- For equity and option orders, types STOP and STOP LIMIT represent the ["Stop On Quote"](#) and ["Stop Limit on Quote"](#) price types offered by E*TRADE, respectively.
- For equity orders, use limit price for type LIMIT, stop price for type STOP, and both prices for type STOP_LIMIT. For option orders, use limit limit price for type LIMIT, stop price for type STOP, and stop-limit price for type STOP_LIMIT.
- The `clientOrderID` element is used to ensure that a duplicate order is not inadvertently submitted. Within a given account, `clientOrderID` must be unique for every order (regardless of which APIs are used).

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Preview changes	Load a display with order details from the List Orders API. Display also shows account ID, name, and balances, as well as product fundamentals, intraday, and current (streaming) price info. In a loop, whenever the user clicks to preview the estimated cost, call the preview API, reload the display from the response, and allow the user to continue editing, commit, or quit. The application should always validate all user inputs and also check the response for error messages.	Get Quote, List Accounts, Get Account Balance, Streaming API

Related APIs

Place Equity Order, Preview Equity Order Change, Preview Option Order, Get Quote, List Accounts, Get Account Balance

Sandbox Samples

The following is an example of a request and response in the sandbox environment. Note that the HTTP POST method is used.

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/previewchangeequityorder.json
```

Request Parameters - JSON

```
{
```

```

"previewChangeEquityOrder":{
  "-xmlns":"http://order.etws.etrade.com",
  "changeEquityOrderRequest":{
    "priceType":"","
    "orderTerm":"GOOD_UNTIL_CANCEL",
    "accountId":"83550325",
    "orderNum":"162",
    "clientOrderId":"asdf1234",
    "limitPrice":"","
    "previewId":"","
    "stopPrice":"","
    "allOrNone":"","
    "quantity":"","
    "reserveOrder":"","
    "reserveQuantity":""
  }
}
}

```

Response

```

{
  "previewChangeEquityOrderResponse":{
    "equityOrderResponse":{
      "accountId":83405188,
      "allOrNone":false,
      "estimatedCommission":5,
      "estimatedTotalAmount":77,
      "orderNum":0,
      "orderTime":1267488794333,
      "previewTime":1267488833223,
      "previewId":499501471087,
      "quantity":4,
      "reserveOrder":false,
      "reserveQuantity":0,
      "orderTerm":"GOOD_FOR_DAY",
      "limitPrice":16,
      "stopPrice":15,
      "symbolDesc":"TD AMERITRADE HLDG CORP COM",
      "symbol":"AMTD",
      "orderAction":"BUY",
      "priceType":"STOP_LIMIT"
    }
  }
}

```

Place Equity Order Change

Submits changes to an equity order request.

Description

This API submits changes to an equity order - essentially cancelling the original order and creating a new one, with a new order ID which appears in the XML response. In a typical workflow, the order is first previewed with Preview Change Equity Order.

Throughout this documentation, the terms "submit" and "place" refer to entering an order into the system to be executed. The actual execution of the order depends on a variety of circumstances. To receive timely notification of the order status, use the streaming order status API, detailed elsewhere in this documentation.

The related Preview Equity Order Change API returns a preview of an order change that includes estimated order cost and a unique preview ID. It is possible to submit an order change without first previewing it, but a typical workflow lets the user preview the order change first. Be aware that, once the change has been previewed, submitting it requires that it include the preview ID and the same parameters that were used in the preview.

URL

<https://etws.etrade.com/order/rest/placechangeequityorder>

HTTP Method: POST

Since this is a POST request, the parameters are included in the request as XML or JSON.

Request Parameters

Parameter	Type	Required?	Description
accountId	integer	required	Numeric account ID
orderNum	integer	required	Order number that identifies the order to be changed
clientOrderId	string	optional	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
previewId	long	conditional	If the change was not previewed, this parameter should not be specified. If the change was previewed, this parameter must specify the numeric preview ID from the preview, and other parameters of this request must match the parameters of the preview.

priceType	enum	required	The type of pricing specified in the equity order. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE If STOP, requires a stopPrice. If LIMIT, requires a limitPrice. If STOP_LIMIT, requires both.
limitPrice	double	conditional	The highest price at which to buy or the lowest price at which to sell. Required if priceType is LIMIT.
stopPrice	double	conditional	The price at which to buy or sell if specified in a stop order. Required if priceType is STOP.
allOrNone	boolean	optional	If TRUE, the transactions specified in the order must be executed all at once, or not at all. Default is FALSE.
quantity	integer	required	The number of shares to buy or sell
reserveOrder	boolean	optional	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. Default is FALSE. If TRUE, must also specify the reserveQuantity.
reserveQuantity	integer	conditional	The number of shares displayed for a reserve order. This parameter is required if reserveOrder is TRUE.
orderTerm	enum	required	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)

Response Properties

Property	Type	Description
accountId	integer	Numeric account ID
allOrNone	boolean	If TRUE, the transactions specified in the order must be executed all at once, or not at all.
estimatedCommission	double	The cost billed to the user to perform the requested action
estimatedTotalAmount	double	The cost or proceeds, including broker commission, resulting from the requested action
messageList	complex	Container for messages describing the result of the action
message	complex	Container for a result message
msgDesc	string	Text of the result message, indicating order status, success or failure, additional requirements that must be met before placing the order, etc. Applications typically display this message to the user, which may result in further user action.

msgCode	integer	Standard numeric code of the result message. Refer to the Error Messages documentation for examples. May optionally be displayed to the user, but is primarily intended for internal use.
orderNum	integer	Numeric ID for this order
orderTime	long	The time the order was submitted, in epoch time
previewTime	long	The time of the preview referenced in the change request, if any
previewId	long	Numeric preview ID
quantity	integer	The number of shares to buy or sell
reserveOrder	boolean	If TRUE, this is a reserve order - meaning that only a limited number of shares will be publicly displayed, instead of the entire order, to avoid influencing other traders.
reserveQuantity	integer	The number of shares to be publicly displayed if this is a reserve order
orderTerm	string	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
priceType	string	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE
limitPrice	double	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Returned if priceType is LIMIT.
stopPrice	double	The price at which a stock is to be bought or sold if specified in a stop order. Returned if priceType is STOP.
symbolDesc	string	Text description of the security being bought or sold
symbol	string	The market symbol for the security being bought or sold
orderAction	string	The action that the broker is requested to perform. Possible values are: <ul style="list-style-type: none"> • BUY • SELL • BUY_OPEN • SELL_OPEN

Sample Request

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/placechangeequityorder
```

Request Parameters - XML

```
<placeChangeEquityOrder xmlns="http://order.etws.etrade.com">
  <changeEquityOrderRequest>
    <accountId>83405188</accountId>
    <orderNum>14</orderNum>
    <clientId></clientId>
    <previewId></previewId>
    <limitPrice>5</limitPrice>
    <stopPrice></stopPrice>
    <allOrNone></allOrNone>
    <quantity></quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
    <priceType>LIMIT</priceType>
    <orderTerm></orderTerm>
  </changeEquityOrderRequest>
</placeChangeEquityOrder>
```

Request Parameters - JSON

```
{
  "placeChangeEquityOrder": {
    "-xmlns": "http://order.etws.etrade.com",
    "changeEquityOrderRequest": {
      "accountId": "83405188",
      "orderNum": "14",
      "limitPrice": "5",
      "priceType": "LIMIT"
    }
  }
}
```

Sample response - XML

```
<PlaceChangeEquityOrderResponse>
  <equityOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>7.99</estimatedCommission>
    <estimatedTotalAmount>27.99</estimatedTotalAmount>
    <messageList>
      <message>
        <msgDesc>
          Your order was successfully entered during market hours.
        </msgDesc>
        <msgCode>1026</msgCode>
      </message>
    </messageList>
    <orderNum>15</orderNum>
    <orderTime>1269424985364</orderTime>
    <previewTime>0</previewTime>
    <previewId>0</previewId>
```

```

    <quantity>4</quantity>
    <reserveOrder>false</reserveOrder>
    <reserveQuantity>0</reserveQuantity>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <limitPrice>5</limitPrice>
    <symbolDesc>E TRADE FINANCIAL CORP COM</symbolDesc>
    <symbol>ETFC</symbol>
    <orderAction>BUY</orderAction>
    <priceType>LIMIT</priceType>
  </equityOrderResponse>
</PlaceChangeEquityOrderResponse>

```

Sample response – JSON

```

{
  "PlaceChangeEquityOrderResponse": {
    "equityOrderResponse": {
      "accountId": "83405188",
      "allOrNone": "FALSE",
      "estimatedCommission": "7.99",
      "estimatedTotalAmount": "27.99",
      "messageList": {
        "message": {
          "msgDesc": "Your order was successfully entered during market hours.",
          "msgCode": "1026"
        }
      },
      "orderNum": "15",
      "orderTime": "1269424985364",
      "previewTime": "0",
      "previewId": "0",
      "quantity": "4",
      "reserveOrder": "FALSE",
      "reserveQuantity": "0",
      "orderTerm": "GOOD_FOR_DAY",
      "limitPrice": "5",
      "symbolDesc": "E TRADE FINANCIAL CORP COM",
      "symbol": "ETFC",
      "orderAction": "BUY",
      "priceType": "LIMIT"
    }
  }
}

```

Notes

- For equity and option orders, types STOP and STOP LIMIT represent the ["Stop On Quote"](#) and ["Stop Limit on Quote"](#) price types offered by E*TRADE, respectively.
- For equity orders, use limit price for type LIMIT, stop price for type STOP, and both prices for type STOP_LIMIT. For option orders, use limit limit price for type LIMIT, stop price for type STOP, and stop-limit price for type STOP_LIMIT.

- The `clientOrderID` element is used to ensure that a duplicate order is not inadvertently submitted. Within a given account, `clientOrderID` must be unique for every order (regardless of which APIs are used).

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Change order	On the order screen, enable the menu option to place an order only when displaying the order preview. In addition to the order preview (and optionally the original order), show account ID, name, and balances, as well as product fundamentals, intraday, and current (streaming) price info. Confirm that balance is sufficient to cover estimated cost before committing the change. Display response.	Get Quote, List Accounts, Get Account Balance, Streaming API
Display order status	When placing or changing an order, register for push notifications on the account. Have a listener process on each page appropriately - for instance, be prepared to display order status updates on the completed order display, order list display, or account display.	<i>See Notifications and Streaming API documentation for details.</i>

Related APIs

Preview Equity Order Change, Place Equity Order, Place Option Order

Sandbox Samples

The following is an example of a request and response in the sandbox environment. Note that the HTTP POST method is used.

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/placechangeequityorder
```

Request Parameters - XML

```
<placeChangeEquityOrder xmlns="http://order.etws.etrade.com">
  changeEquityOrderRequest>
    <priceType></priceType>
    <orderTerm>GOOD_UNTIL_CANCEL</orderTerm>
    <accountId>83550325</accountId>
    <orderNum>162</orderNum>
    <clientOrderId>asdf1234</clientOrderId>
    <limitPrice></limitPrice>
    <previewId></previewId>
    <stopPrice></stopPrice>
    <allOrNone></allOrNone>
```



```

    <quantity></quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
  </changeEquityOrderRequest>
</placeChangeEquityOrder>

```

Response

```

<PlaceChangeEquityOrderResponse>
  <equityOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>5</estimatedCommission>
    <estimatedTotalAmount>77</estimatedTotalAmount>
    <messageList>
      <message>
        <msgDesc>Your order was successfully entered during market hours.</msgDesc>
        <msgCode>1026</msgCode>
      </message>
    </messageList>
    <orderNum>1320</orderNum>
    <orderTime>1267487586056</orderTime>
    <previewTime>0</previewTime>
    <previewId>0</previewId>
    <quantity>4</quantity>
    <reserveOrder>false</reserveOrder>
    <reserveQuantity>0</reserveQuantity>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <limitPrice>16</limitPrice>
    <stopPrice>15</stopPrice>
    <symbolDesc>TD AMERITRADE HLDG CORP COM</symbolDesc>
    <symbol>AMTD</symbol>
    <orderAction>BUY</orderAction>
    <priceType>STOP_LIMIT</priceType>
  </equityOrderResponse>
</PlaceChangeEquityOrderResponse>

```

Preview Option Order

Returns a preview of an option order.

Description

This API accepts a simulated option order as input and returns a preview of the order, with estimated costs, allowing the user to review the order before placing it.

The preview response includes a preview ID number which must be referenced when placing the actual order.

URL

<https://etws.etrade.com/order/rest/previewoptionorder>

HTTP Method: POST

Since this is a POST request, the parameters are included in the request as XML or JSON.

Request Parameters

Parameter	Type	Required?	Description
accountId	integer	required	Numeric account ID
symbolInfo	complex	required	Container for symbol information
symbol	string	required	The market symbol for the underlier
callOrPut	enum	required	Option type - specifies either CALL or PUT
strikePrice	double	required	The strike price for this option
expirationYear	integer	required	The 4-digit year the option will expire
expirationMonth	integer	required	The month (1-12) the option will expire
expirationDay	integer	required	The day (1-31) the option will expire
orderAction	enum	required	The action that the broker is requested to perform. Possible values are: <ul style="list-style-type: none">• BUY_OPEN• SELL_OPEN• BUY_CLOSE• SELL_CLOSE
priceType	enum	required	The type of pricing specified in the equity order. Possible values are: <ul style="list-style-type: none">• MARKET• LIMIT• STOP• STOP_LIMIT If STOP, requires a stopPrice. If LIMIT, requires a limitPrice. If STOP_LIMIT, requires a stopLimitPrice.

limitPrice	double	conditional	The highest price at which to buy or the lowest price at which to sell. Required if <code>priceType</code> is LIMIT.
stopPrice	double	conditional	The price at which to buy or sell if specified in a stop order. Required if <code>priceType</code> is STOP.
stopLimitPrice	double	conditional	The designated price for a stop-limit order. Required if <code>priceType</code> is STOP_LIMIT.
allOrNone	boolean	optional	If TRUE, the transactions specified in the order must be executed all at once, or not at all. Default is FALSE.
quantity	integer	required	The number of shares to buy or sell
reserveOrder	boolean	optional	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. Default is FALSE. If TRUE, must also specify the <code>reserveQuantity</code> .
reserveQuantity	integer	conditional	The number of shares displayed for a reserve order. Required if <code>reserveOrder</code> is TRUE.
orderTerm	enum	required	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
routingDestination	enum	optional	The exchange where the order should be executed. Users may want to specify this if they believe they can get a better order fill at a specific exchange, rather than relying on the automatic order routing system. Possible values are: <ul style="list-style-type: none"> • AUTO (default) • ARCA • NSDQ • NYSE
clientOrderId	string	optional	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
routingDestination	enum	optional	The exchange where the order should be executed. Possible values are: <ul style="list-style-type: none"> • AUTO (default) • ARCA • NSDQ • NYSE

Response Properties

Property	Type	Description
accountId	integer	Numeric account ID

allOrNone	boolean	If TRUE, the transactions specified in the order must be executed all at once, or not at all.
estimatedCommission	double	The cost billed to the user to perform the requested action
estimatedTotalAmount	double	The cost or proceeds, including broker commission, resulting from the requested action
previewTime	long	The time of this preview, in epoch time
previewId	long	Numeric preview ID
optionSymbol	complex	Container for the option identifier
symbol	string	The market symbol for the underlier
callOrPut	string	Option type - specifies either CALL or PUT
strikePrice	double	The strike price for this option
expirationYear	integer	The 4-digit year the option will expire
expirationMonth	integer	The month (1-12) the option will expire
expirationDay	integer	The day (1-31) the option will expire
quantity	integer	The number of shares to buy or sell
reserveOrder	boolean	If TRUE, this is a reserve order - meaning that only a limited number of shares will be publicly displayed, instead of the entire order, to avoid influencing other traders.
reserveQuantity	integer	The number of shares to be publicly displayed if this is a reserve order
orderTerm	string	Specifies the term for which the order is in effect. Possible values are: • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
orderAction	string	The action that the broker is requested to perform. Possible values are: • BUY_OPEN • SELL_OPEN • BUY_CLOSE • SELL_CLOSE
priceType	string	The type of pricing. Possible values are: • MARKET • LIMIT • STOP • STOP_LIMIT
limitPrice	double	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Returned if priceType is LIMIT.
stopPrice	double	The price at which a stock is to be bought or sold if specified in a stop order. Returned if priceType is STOP.
stopLimitPrice	double	The designated price for a stop-limit order. Returned if priceType is STOP_LIMIT.

routingDestination	string	The exchange where the user requests that the order be executed. Possible values are: <ul style="list-style-type: none"> • AUTO • ARCA • NSDQ • NYSE
messageList	complex	Container for messages describing the result of the action
message	complex	Container for a result message
msgDesc	string	Text of the result message, indicating order status, success or failure, additional requirements that must be met before placing the order, etc. Applications typically display this message to the user, which may result in further user action.
msgCode	integer	Standard numeric code of the result message. Refer to the Error Messages documentation for examples. May optionally be displayed to the user, but is primarily intended for internal use.

Sample Request

Request URL

POST <https://etwssandbox.etrade.com/order/sandbox/rest/previewoptionorder>

Request Parameters - XML

```
<PreviewOptionOrder xmlns="http://order.etws.etrade.com">
  <OptionOrderRequest>
    <accountId>83405188</accountId>
    <limitPrice></limitPrice>
    <stopPrice></stopPrice>
    <stopLimitPrice></stopLimitPrice>
    <allOrNone></allOrNone>
    <quantity>1</quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
    <symbolInfo>
      <symbol>IBM</symbol>
      <callOrPut>CALL</callOrPut>
      <strikePrice>115</strikePrice>
      <expirationYear>2010</expirationYear>
      <expirationMonth>4</expirationMonth>
      <expirationDay>17</expirationDay>
    </symbolInfo>
    <orderAction>BUY_OPEN</orderAction>
    <priceType>MARKET</priceType>
    <routingDestination></routingDestination>
    <marketSession></marketSession>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
  </OptionOrderRequest>
</PreviewOptionOrder>
```

Request Parameters - JSON

```
{
  "PreviewOptionOrder": {
    "-xmlns": "http://order.etws.etrade.com",
    "OptionOrderRequest": {
      "accountId": "83405188",
      "quantity": "1",
      "symbolInfo": {
        "symbol": "IBM",
        "callOrPut": "CALL",
        "strikePrice": "115",
        "expirationYear": "2010",
        "expirationMonth": "4",
        "expirationDay": "17"
      },
      "orderAction": "BUY_OPEN",
      "priceType": "MARKET",
      "orderTerm": "GOOD_FOR_DAY"
    }
  }
}
```

Sample response - XML

```
<PreviewOptionOrderResponse>
  <optionOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>8.74</estimatedCommission>
    <estimatedTotalAmount>1463.78</estimatedTotalAmount>
    <previewTime>1269430877665</previewTime>
    <previewId>449548422022</previewId>
    <quantity>1</quantity>
    <reserveOrder>false</reserveOrder>
    <reserveQuantity>0</reserveQuantity>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <optionSymbol>
      <symbol>IBM</symbol>
      <callOrPut>CALL</callOrPut>
      <expirationYear>2010</expirationYear>
      <expirationMonth>4</expirationMonth>
      <expirationDay>17</expirationDay>
    </optionSymbol>
    <orderAction>BUY_OPEN</orderAction>
    <priceType>MARKET</priceType>
  </optionOrderResponse>
</PreviewOptionOrderResponse>
```

Sample response – JSON

```
{
  "PreviewOptionOrderResponse": {
```

```

    "optionOrderResponse": {
      "accountId": "83405188",
      "allOrNone": "FALSE",
      "estimatedCommission": "8.74",
      "estimatedTotalAmount": "1463.78",
      "previewTime": "1269430877665",
      "previewId": "449548422022",
      "quantity": "1",
      "reserveOrder": "FALSE",
      "reserveQuantity": "0",
      "orderTerm": "GOOD_FOR_DAY",
      "optionSymbol": {
        "symbol": "IBM",
        "callOrPut": "CALL",
        "expirationYear": "2010",
        "expirationMonth": "4",
        "expirationDay": "17"
      },
      "orderAction": "BUY_OPEN",
      "priceType": "MARKET"
    }
  }
}

```

Notes

- For equity and option orders, types STOP and STOP LIMIT represent the ["Stop On Quote"](#) and ["Stop Limit on Quote"](#) price types offered by E*TRADE, respectively.
- For equity orders, use limit price for type LIMIT, stop price for type STOP, and both prices for type STOP_LIMIT. For option orders, use limit limit price for type LIMIT, stop price for type STOP, and stop-limit price for type STOP_LIMIT.
- The `clientOrderID` element is used to ensure that a duplicate order is not inadvertently submitted. Within a given account, `clientOrderID` must be unique for every order (regardless of which APIs are used).

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Order preview	User interacts with screen to assemble an order by selecting an account to use and a product to trade. Display shows account ID, name, and balances, as well as product fundamentals, intraday, and current (streaming) price info. In a loop, whenever the user clicks to preview the estimated cost, call the preview API, reload the display from the response, and allow the user to continue editing, commit, or quit. The application should always validate all user inputs and also check the response for error messages.	Get Quote, List Accounts, Get Account Balance, Streaming API

Related APIs

Place Option Order, Preview Option Order Change, Preview Equity Order, Get Quote, List Accounts, Get Account Balance

Sandbox Samples

The following is an example of a request and response in the sandbox environment. Note that the HTTP POST method is used.

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/previewoptionorder
```

Request Parameters - XML

```
<PreviewOptionOrder xmlns="http://order.etws.etrade.com">
  <OptionOrderRequest>
    <stopLimitPrice></stopLimitPrice>
    <symbolInfo>
      <symbol>AAPL</symbol>
      <callOrPut>CALL</callOrPut>
      <strikePrice>585</strikePrice>
      <expirationYear>2012</expirationYear>
      <expirationMonth>07</expirationMonth>
      <expirationDay>21</expirationDay>
    </symbolInfo>
    <orderAction>BUY_OPEN</orderAction>
    <priceType>LIMIT</priceType>
    <routingDestination></routingDestination>
    <marketSession>REGULAR</marketSession>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <allOrNone></allOrNone>
    <quantity>4</quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
    <accountId>83550325</accountId>
    <clientOrderId>12345</clientOrderId>
    <limitPrice>3</limitPrice>
    <previewId></previewId>
    <stopPrice></stopPrice>
  </OptionOrderRequest>
</PreviewOptionOrder>
```

Response

```
<PreviewOptionOrderResponse>
  <optionOrderResponse>
    <accountId>83405188</accountId>
```



```
<allOrNone>false</allOrNone>
<estimatedCommission>8</estimatedCommission>
<estimatedTotalAmount>12008.056</estimatedTotalAmount>
<previewTime>1240983475632</previewTime>
<previewId>449294315087</previewId>
<quantity>4</quantity>
<reserveOrder>false</reserveOrder>
<reserveQuantity>0</reserveQuantity>
<orderTerm>FILL_OR_KILL</orderTerm>
<limitPrice>30</limitPrice>
<optionSymbol>
  <symbol>MSFT</symbol>
  <callOrPut>CALL</callOrPut>
  <strikePrice>12.500000</strikePrice>
  <expirationYear>2010</expirationYear>
  <expirationMonth>4</expirationMonth>
  <expirationDay>17</expirationDay>
</optionSymbol>
<orderAction>BUY_OPEN</orderAction>
<priceType>LIMIT</priceType>
</optionOrderResponse>
</PreviewOptionOrderResponse>
```

Place Option Order

Submits an option order.

Description

This API places an option order. In a typical workflow, the order is first previewed with Preview Option Order. Once it has been placed, the order can be viewed with the List Orders API.

Throughout this documentation, the terms "submit" and "place" refer to entering an order into the system to be executed. The actual execution of the order depends on a variety of circumstances. To receive timely notification of the order status, use the streaming order status API, detailed elsewhere in this documentation.

The related Preview Option Order API returns a preview of an order that includes estimated order cost and a unique preview ID. It is possible to place an order without first previewing it, but a typical order workflow lets the user preview the order before submitting it. Be aware that, once an order has been previewed, placing the order requires that it include the preview ID and the same parameters that were used in the preview.

URL

<https://etws.etrade.com/order/rest/placeoptionorder>

HTTP Method: POST

Since this is a POST request, the parameters are included in the request as XML or JSON.

Request Parameters

Parameter	Type	Required?	Description
accountId	integer	required	Numeric account ID
symbolInfo	complex	required	Container for option identifier
symbol	string	required	The market symbol for the underlier.
callOrPut	enum	required	Option type - specifies either CALL or PUT
strikePrice	double	required	The strike price for this option
expirationYear	integer	required	The 4-digit year the option will expire
expirationMonth	integer	required	The month (1-12) the option will expire
expirationDay	integer	required	The day (1-31) the option will expire
orderAction	enum	required	The action that the broker is requested to perform. Possible values are: <ul style="list-style-type: none">• BUY_OPEN• SELL_OPEN• BUY_CLOSE• SELL_CLOSE

priceType	enum	required	The type of pricing specified in the equity order. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT If STOP, requires a stopPrice. If LIMIT, requires a limitPrice. If STOP_LIMIT, requires a stopLimitPrice.
limitPrice	double	conditional	The highest price at which to buy or the lowest price at which to sell. Required if priceType is STOP or STOP_LIMIT.
stopPrice	double	conditional	The price at which to buy or sell if specified in a stop order. Required if priceType is STOP.
stopLimitPrice	double	conditional	The designated boundary price for a stop-limit order. Required if priceType is STOP_LIMIT.
allOrNone	boolean	optional	If TRUE, the transactions specified in the order must be executed all at once, or not at all. Default is FALSE.
quantity	integer	required	The number of shares to buy or sell
reserveOrder	boolean	optional	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. Default is FALSE. If TRUE, must also specify the reserveQuantity.
reserveQuantity	integer	conditional	The number of shares displayed for a reserve order. Required if reserveOrder is TRUE.
orderTerm	enum	required	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
routingDestination	enum	optional	The exchange where the order should be executed. Users may want to specify this if they believe they can get a better order fill at a specific exchange, rather than relying on the automatic order routing system. Possible values are: <ul style="list-style-type: none"> • AUTO (default) • ARCA • NSDQ • NYSE
previewId	long	conditional	If the order was not previewed, this parameter should not be specified. If the order was previewed, this parameter must specify the numeric preview ID from the preview, and other parameters of this request must match the parameters of the preview.

clientOrderId	string	required	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
routingDestination	enum	optional	The exchange where the order should be executed. Possible values are: <ul style="list-style-type: none"> • AUTO (default) • ARCA • NSDQ • NYSE

Response Properties

Property	Type	Description
accountId	integer	Numeric account ID
allOrNone	boolean	If TRUE, the transactions specified in the order must be executed all at once, or not at all.
estimatedCommission	double	The cost billed to the user to perform the requested action
estimatedTotalAmount	double	The cost or proceeds, including broker commission, resulting from the requested action
messageList	complex	Container for messages describing the result of the action
message	complex	Container for a result message
msgDesc	string	Text of the result message, indicating order status, success or failure, additional requirements that must be met before placing the order, etc. Applications typically display this message to the user, which may result in further user action.
msgCode	integer	Standard numeric code of the result message. Refer to the Error Messages documentation for examples. May optionally be displayed to the user, but is primarily intended for internal use.
orderNum	integer	Numeric order ID
orderTime	long	The time the order was submitted
optionSymbol	complex	Container for the option identifier
symbol	string	The market symbol for the underlier
callOrPut	string	Option type - specifies either CALL or PUT
strikePrice	double	The strike price for this option
expirationYear	integer	The 4-digit year the option will expire
expirationMonth	integer	The month (1-12) the option will expire
expirationDay	integer	The day (1-31) the option will expire
quantity	integer	The number of shares to buy or sell
reserveOrder	boolean	If TRUE, this is a reserve order - meaning that only a limited number of shares will be publicly displayed, instead of the entire order, to avoid influencing other traders.
reserveQuantity	integer	The number of shares to be publicly displayed if this is a reserve order

orderTerm	string	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
orderAction	string	The action that the broker is requested to perform. Possible values are: <ul style="list-style-type: none"> • BUY_OPEN • SELL_OPEN • BUY_CLOSE • SELL_CLOSE
priceType	string	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT
limitPrice	double	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Returned if priceType is LIMIT.
stopPrice	double	The price at which a stock is to be bought or sold if specified in a stop order. Returned if priceType is STOP.
stopLimitPrice	double	The designated boundary price for a stop-limit order. Returned if priceType is STOP_LIMIT.
routingDestination	string	The exchange where the order should be executed. Users may want to specify this if they believe they can get a better order fill at a specific exchange, rather than relying on the automatic order routing system. Possible values are: <ul style="list-style-type: none"> • AUTO • ARCA • NSDQ • NYSE

Sample Request

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/placeoptionorder.json
```

Request Parameters - XML

```
<PlaceOptionOrder xmlns="http://order.etws.etrade.com">
  <OptionOrderRequest>
    <accountId>83405188</accountId>
    <clientOrderId>259</clientOrderId>
    <limitPrice>10</limitPrice>
    <previewId></previewId>
    <stopPrice></stopPrice>
    <stopLimitPrice></stopLimitPrice>
  </OptionOrderRequest>
</PlaceOptionOrder>
```

```

    <allOrNone></allOrNone>
    <quantity>1</quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
    <symbolInfo>
      <symbol>IBM</symbol>
      <callOrPut>CALL</callOrPut>
      <strikePrice>115</strikePrice>
      <expirationYear>2010</expirationYear>
      <expirationMonth>4</expirationMonth>
      <expirationDay>17</expirationDay>
    </symbolInfo>
    <orderAction>BUY_OPEN</orderAction>
    <priceType>LIMIT</priceType>
    <routingDestination></routingDestination>
    <marketSession></marketSession>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
  </OptionOrderRequest>
</PlaceOptionOrder>

```

Request Parameters - JSON

```

{
  "PlaceOptionOrder": {
    "-xmlns": "http://order.etws.etrade.com",
    "OptionOrderRequest": {
      "accountId": "83405188",
      "clientOrderId": "259",
      "limitPrice": "10",
      "quantity": "1",
      "symbolInfo": {
        "symbol": "IBM",
        "callOrPut": "CALL",
        "strikePrice": "115",
        "expirationYear": "2010",
        "expirationMonth": "4",
        "expirationDay": "17"
      },
      "orderAction": "BUY_OPEN",
      "priceType": "LIMIT",
      "orderTerm": "GOOD_FOR_DAY"
    }
  }
}

```

Sample response - XML

```

<PlaceOptionOrderResponse>
  <optionOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>8.74</estimatedCommission>
    <estimatedTotalAmount>1008.78</estimatedTotalAmount>
  </optionOrderResponse>
</PlaceOptionOrderResponse>

```

```

<messageList>
  <message>
    <msgDesc>Your order was successfully entered during market hours.</msgDesc>
    <msgCode>1026</msgCode>
  </message>
</messageList>
<orderNum>257</orderNum>
<orderTime>1269430628956</orderTime>
<quantity>1</quantity>
<reserveOrder>false</reserveOrder>
<reserveQuantity>0</reserveQuantity>
<orderTerm>GOOD_FOR_DAY</orderTerm>
<limitPrice>10</limitPrice>
<optionSymbol>
  <symbol>IBM</symbol>
  <callOrPut>CALL</callOrPut>
  <strikePrice>115.000000</strikePrice>
  <expirationYear>2010</expirationYear>
  <expirationMonth>4</expirationMonth>
  <expirationDay>17</expirationDay>
</optionSymbol>
<orderAction>BUY_OPEN</orderAction>
<priceType>LIMIT</priceType>
</optionOrderResponse>
</PlaceOptionOrderResponse>

```

Sample response – JSON

```

{
  "PlaceOptionOrderResponse": {
    "optionOrderResponse": {
      "accountId": "83405188",
      "allOrNone": "FALSE",
      "estimatedCommission": "8.74",
      "estimatedTotalAmount": "1008.78",
      "messageList": {
        "message": {
          "msgDesc": "Your order was successfully entered during market hours.",
          "msgCode": "1026"
        }
      },
    },
    "orderNum": "257",
    "orderTime": "1269430628956",
    "quantity": "1",
    "reserveOrder": "FALSE",
    "reserveQuantity": "0",
    "orderTerm": "GOOD_FOR_DAY",
    "limitPrice": "10",
    "optionSymbol": {
      "symbol": "IBM",
      "callOrPut": "CALL",
      "strikePrice": "115.000000",
      "expirationYear": "2010",
    },
  },
}

```

```

        "expirationMonth": "4",
        "expirationDay": "17"
    },
    "orderAction": "BUY_OPEN",
    "priceType": "LIMIT"
}
}
}

```

Notes

- For equity and option orders, types STOP and STOP LIMIT represent the ["Stop On Quote"](#) and ["Stop Limit on Quote"](#) price types offered by E*TRADE, respectively.
- For equity orders, use limit price for type LIMIT, stop price for type STOP, and both prices for type STOP_LIMIT. For option orders, use limit limit price for type LIMIT, stop price for type STOP, and stop-limit price for type STOP_LIMIT.
- The `clientOrderID` element is used to ensure that a duplicate order is not inadvertently submitted. Within a given account, `clientOrderID` must be unique for every order (regardless of which APIs are used).

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Place order	On the order screen, enable the menu option to place an order only when displaying the response from the preview API. In addition to the order preview, show account ID, name, and balances, as well as product fundamentals, intraday, and current (streaming) price info. Confirm that the user account is approved for option trading. Check market rules for option orders and display warnings if needed. Display response.	Get Quote, List Accounts, Get Account Balance, Streaming API
Display order status	When placing or changing an order, register for push notifications on the account. Have a listener process on each page appropriately - for instance, be prepared to display order status updates on the completed order display, order list display, or account display.	<i>See Notifications and Streaming API documentation for details.</i>

Related APIs

Preview Option Order, Preview Option Order Change, Place Equity Order

Sandbox Samples

The following is an example of a request and response in the sandbox environment. Note that the HTTP POST method is used.

Request URL

POST <https://etwssandbox.etrade.com/order/sandbox/rest/placeoptionorder>

Request Parameters - XML

```
<PlaceOptionOrder xmlns="http://order.etws.etrade.com">
  <OptionOrderRequest>
    <stopLimitPrice></stopLimitPrice>
    <symbolInfo>
      <symbol>AAPL</symbol>
      <callOrPut>CALL</callOrPut>
      <strikePrice>585</strikePrice>
      <expirationYear>2012</expirationYear>
      <expirationMonth>07</expirationMonth>
      <expirationDay>21</expirationDay>
    </symbolInfo>
    <orderAction>BUY_OPEN</orderAction>
    <priceType>LIMIT</priceType>
    <routingDestination></routingDestination>
    <marketSession>REGULAR</marketSession>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <allOrNone></allOrNone>
    <quantity>4</quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
    <accountId>83550325</accountId>
    <clientOrderId>12345</clientOrderId>
    <limitPrice>3</limitPrice>
    <previewId></previewId>
    <stopPrice></stopPrice>
  </OptionOrderRequest>
</PlaceOptionOrder>
```

Response

```
<PlaceOptionOrderResponse>
  <optionOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>8</estimatedCommission>
    <estimatedTotalAmount>12008.056</estimatedTotalAmount>
    <messageList>
      <message>
        <msgDesc>Your order was successfully entered during market hours.</msgDesc>
        <msgCode>1026</msgCode>
      </message>
    </messageList>
    <orderNum>1260</orderNum>
    <orderTime>1267042264205</orderTime>
    <quantity>4</quantity>
    <reserveOrder>false</reserveOrder>
    <reserveQuantity>0</reserveQuantity>
```

```
<orderTerm>FILL_OR_KILL</orderTerm>
<limitPrice>30</limitPrice>
<optionSymbol>
  <symbol>MSFT</symbol>
  <callOrPut>CALL</callOrPut>
  <strikePrice>12.500000</strikePrice>
  <expirationYear>2010</expirationYear>
  <expirationMonth>4</expirationMonth>
  <expirationDay>17</expirationDay>
</optionSymbol>
<orderAction>BUY_OPEN</orderAction>
<priceType>LIMIT</priceType>
</optionOrderResponse>
</PlaceOptionOrderResponse>
```

Preview Option Order Change

Returns a preview of a change to an option order.

Description

This API accepts a simulated option order change as input and returns a preview of the modified order, with estimated costs, allowing the user to review the change before submitting it.

The preview response includes a preview ID number which must be referenced when submitting the actual change request.

URL

<https://etws.etrade.com/order/rest/previewchangeoptionorder>

HTTP Method: POST

Since this is a POST request, the parameters are included in the request as XML or JSON.

Request Parameters

Parameter	Type	Required?	Description
accountId	integer	required	Numeric account ID
orderNum	integer	required	Order number, taken from a previous response, that identifies the order to be changed
clientOrderId	string	optional	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
priceType	enum	required	The type of pricing specified in the equity order. Possible values are: <ul style="list-style-type: none">• MARKET• LIMIT• STOP• STOP_LIMIT If STOP, requires a <code>stopPrice</code> . If LIMIT, requires a <code>limitPrice</code> . If STOP_LIMIT option order, requires a <code>stopLimitPrice</code> .
limitPrice	double	conditional	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Required if <code>priceType</code> is LIMIT.
stopPrice	double	conditional	The price at which to buy or sell if specified in a stop order. Required if <code>priceType</code> is STOP.
stopLimitPrice	double	conditional	The designated price for a stop-limit order. Required if <code>priceType</code> is STOP_LIMIT.

allOrNone	boolean	optional	If TRUE, the transactions specified in the order must be executed all at once, or not at all. Default is FALSE.
quantity	integer	required	The number of shares to buy or sell
reserveOrder	boolean	optional	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. Default is FALSE. If TRUE, must also specify the <code>reserveQuantity</code> .
reserveQuantity	integer	conditional	The number of shares displayed for a reserve order. Required if <code>reserveOrder</code> is TRUE.
orderTerm	enum	required	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)

Response Properties

Property	Type	Description
accountId	integer	Numeric account ID
allOrNone	boolean	If TRUE, the transactions specified in the order are to be executed all at once, or not at all.
estimatedCommission	double	The cost billed to the user to perform the requested action.
estimatedTotalAmount	double	The cost or proceeds, including broker commission, resulting from the requested action.
optionSymbol	complex	Container for the option identifier
symbol	string	The market symbol for the underlier
callOrPut	string	Option type - specifies either CALL or PUT
strikePrice	double	The strike price for this option
expirationYear	integer	The 4-digit year the option will expire
expirationMonth	integer	The month (1-12) the option will expire
expirationDay	integer	The day (1-31) the option will expire
quantity	integer	The number of shares to buy or sell
reserveOrder	boolean	If TRUE, this is a reserve order - meaning that only a limited number of shares will be publicly displayed, instead of the entire order, to avoid influencing other traders.
reserveQuantity	integer	The number of shares to be publicly displayed if this is a reserve order
orderTerm	string	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)

priceType	string	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE
limitPrice	double	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Returned if priceType is LIMIT.
stopPrice	double	The price at which a stock is to be bought or sold if specified in a stop order. Returned if priceType is STOP.
stopLimitPrice	double	The designated price for a stop-limit order. Returned if priceType is STOP_LIMIT.
orderAction	string	The action that the broker is requested to perform. Possible values are: <ul style="list-style-type: none"> • BUY • SELL • BUY_OPEN • SELL_OPEN
previewTime	long	The time of this preview, in epoch time
previewId	long	Numeric preview ID
messageList	complex	Container for messages describing the result of the action
message	complex	Container for a result message
msgDesc	string	Text of the result message, indicating order status, success or failure, additional requirements that must be met before placing the order, etc. Applications typically display this message to the user, which may result in further user action.
msgCode	integer	Standard numeric code of the result message. Refer to the Error Messages documentation for examples. May optionally be displayed to the user, but is primarily intended for internal use.
routingDestination	string	The exchange where the user requests that the order be executed. Possible values are: <ul style="list-style-type: none"> • AUTO • AMEX • BOX • CBOE • ISE • NOM • NYSE • PHX

Sample Request

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/previewchangeoptionorder
```

Request Parameters - XML

```
<previewChangeOptionOrder xmlns="http://order.etws.etrade.com">
  <changeOptionOrderRequest>
    <accountId>83405188</accountId>
    <orderNum>258</orderNum>
    <limitPrice>25</limitPrice>
    <stopPrice></stopPrice>
    <stopLimitPrice></stopLimitPrice>
    <allOrNone></allOrNone>
    <quantity>4</quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
    <priceType>LIMIT</priceType>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
  </changeOptionOrderRequest>
</previewChangeOptionOrder>
```

Request Parameters - JSON

```
{
  "previewChangeOptionOrder": {
    "-xmlns": "http://order.etws.etrade.com",
    "changeOptionOrderRequest": {
      "accountId": "83405188",
      "orderNum": "258",
      "limitPrice": "25",
      "quantity": "4",
      "priceType": "LIMIT",
      "orderTerm": "GOOD_FOR_DAY"
    }
  }
}
```

Sample response - XML

```
<PreviewChangeOptionOrderResponse>
  <optionOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>10.99</estimatedCommission>
    <estimatedTotalAmount>10056.046</estimatedTotalAmount>
    <messageList/>
    <orderNum>0</orderNum>
    <orderTime>1269431139424</orderTime>
    <previewTime>1269432162454</previewTime>
    <previewId>449548438022</previewId>
    <quantity>4</quantity>
    <reserveOrder>false</reserveOrder>
    <reserveQuantity>0</reserveQuantity>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <limitPrice>25</limitPrice>
    <optionSymbol>
```

```

    <symbol>IBM</symbol>
    <callOrPut>CALL</callOrPut>
    <strikePrice>175.000000</strikePrice>
    <expirationYear>2010</expirationYear>
    <expirationMonth>10</expirationMonth>
    <expirationDay>16</expirationDay>
  </optionSymbol>
  <orderAction>BUY_OPEN</orderAction>
  <priceType>LIMIT</priceType>
</optionOrderResponse>
</PreviewChangeOptionOrderResponse>

```

Sample response – JSON

```

{
  "PreviewChangeOptionOrderResponse": {
    "optionOrderResponse": {
      "accountId": "83405188",
      "allOrNone": "FALSE",
      "estimatedCommission": "10.99",
      "estimatedTotalAmount": "10056.046",
      "orderNum": "0",
      "orderTime": "1269431139424",
      "previewTime": "1269432162454",
      "previewId": "449548438022",
      "quantity": "4",
      "reserveOrder": "FALSE",
      "reserveQuantity": "0",
      "orderTerm": "GOOD_FOR_DAY",
      "limitPrice": "25",
      "optionSymbol": {
        "symbol": "IBM",
        "callOrPut": "CALL",
        "strikePrice": "175.000000",
        "expirationYear": "2010",
        "expirationMonth": "10",
        "expirationDay": "16"
      },
      "orderAction": "BUY_OPEN",
      "priceType": "LIMIT"
    }
  }
}

```

Notes

- For equity and option orders, types STOP and STOP LIMIT represent the ["Stop On Quote"](#) and ["Stop Limit on Quote"](#) price types offered by E*TRADE, respectively.
- For equity orders, use limit price for type LIMIT, stop price for type STOP, and both prices for type STOP_LIMIT. For option orders, use limit limit price for type LIMIT, stop price for type STOP, and stop-limit price for type STOP_LIMIT.

- The `clientOrderID` element is used to ensure that a duplicate order is not inadvertently submitted. Within a given account, `clientOrderID` must be unique for every order (regardless of which APIs are used).

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Preview changes	Load a display with order details from the List Orders API. Display also shows account ID, name, and balances, as well as product fundamentals, intraday, and current (streaming) price info. In a loop, whenever the user clicks to preview the estimated cost, call the preview API, reload the display from the response, and allow the user to continue editing, commit, or quit. The application should always validate all user inputs and also check the response for error messages.	Get Quote, List Accounts, Get Account Balance, Streaming API

Related APIs

Place Option Order Change, Preview Option Order, Preview Equity Order Change, Get Quote, List Accounts, Get Account Balance

Sandbox Samples

The following is an example of a request and response in the sandbox environment. Note that the HTTP POST method is used.

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/previewchangeoptionorder
```

Request Parameters - XML

```
<previewChangeOptionOrder xmlns="http://order.etws.etrade.com">
  <changeOptionOrderRequest>
    <stopLimitPrice></stopLimitPrice>
    <priceType></priceType>
    <orderTerm>GOOD_UNTIL_CANCEL</orderTerm>
    <accountId>83550325</accountId>
    <orderNum>162</orderNum>
    <clientOrderId>asdf1234</clientOrderId>
    <limitPrice></limitPrice>
    <previewId></previewId>
    <stopPrice></stopPrice>
    <allOrNone></allOrNone>
    <quantity></quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
```



```
</changeOptionOrderRequest>
</previewChangeOptionOrder>
```

Response

```
<PreviewChangeOptionOrderResponse>
  <optionOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>6.5</estimatedCommission>
    <estimatedTotalAmount>2206.54</estimatedTotalAmount>
    <orderNum>0</orderNum>
    <orderTime>1267231909065</orderTime>
    <previewTime>1267232061708</previewTime>
    <previewId>499501305087</previewId>
    <quantity>2</quantity>
    <reserveOrder>false</reserveOrder>
    <reserveQuantity>0</reserveQuantity>
    <orderTerm>GOOD_UNTIL_CANCEL</orderTerm>
    <limitPrice>11</limitPrice>
    <optionSymbol>
      <symbol>MSFT</symbol>
      <callOrPut>CALL</callOrPut>
      <strikePrice>12.500000</strikePrice>
      <expirationYear>2010</expirationYear>
      <expirationMonth>4</expirationMonth>
      <expirationDay>17</expirationDay>
    </optionSymbol>
    <orderAction>BUY_CLOSE</orderAction>
    <priceType>LIMIT</priceType>
  </optionOrderResponse>
</PreviewChangeOptionOrderResponse>
```

Place Option Order Change

Submits changes to an option order.

Description

This API submits changes to an option order - essentially cancelling the original order and creating a new one, with a new order ID which appears in the XML response. In a typical workflow, the changes are first previewed with Preview Change Option Order.

Throughout this documentation, the terms "submit" and "place" refer to entering an order into the system to be executed. The actual execution of the order depends on a variety of circumstances. To receive timely notification of the order status, use the streaming order status API, detailed elsewhere in this documentation.

The related Preview Option Order Change API returns a preview of an order change that includes estimated order cost and a unique preview ID. It is possible to submit an order change without first previewing it, but a typical workflow lets the user preview the order change first. Be aware that, once the change has been previewed, submitting it requires that it include the preview ID and the same parameters that were used in the preview.

URL

<https://etws.etrade.com/order/rest/placechangeoptionorder>

HTTP Method: POST

Since this is a POST request, the parameters are included in the request as XML or JSON.

Request Parameters

Parameter	Type	Required?	Description
accountId	integer	required	Numeric account ID
orderNum	integer	required	Order number that identifies the order to be changed
clientOrderId	string	required	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
previewId	long	conditional	If the change was not previewed, this parameter should not be specified. If the change was previewed, this parameter must specify the numeric preview ID from the preview, and other parameters of this request must match the parameters of the preview.

priceType	enum	required	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE If STOP, requires a stopPrice. If LIMIT, requires a limitPrice. If STOP_LIMIT option order, requires a stopLimitPrice.
limitPrice	double	conditional	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Required if priceType is LIMIT.
stopPrice	double	conditional	The price at which to buy or sell if specified in a stop order. Required if priceType is STOP.
stopLimitPrice	double	conditional	The designated price for a stop-limit order. Required if priceType is STOP_LIMIT.
allOrNone	boolean	optional	If TRUE, the transactions specified in the order must be executed all at once, or not at all. Default is FALSE.
quantity	integer	required	The number of shares to buy or sell
reserveOrder	boolean	optional	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. Default is FALSE. If TRUE, must also specify the reserveQuantity.
reserveQuantity	integer	conditional	The number of shares displayed for a reserve order. Required if reserveOrder is TRUE.
orderTerm	enum	required	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)

Response Properties

Property	Type	Description
accountId	integer	Numeric account ID
allOrNone	boolean	If TRUE, the transactions specified in the order must be executed all at once, or not at all.
estimatedCommission	double	The cost billed to the user to perform the requested action
estimatedTotalAmount	double	The cost or proceeds, including broker commission, resulting from the requested action
orderNum	integer	Numeric ID for this order
orderTime	long	The time the order was submitted, in epoch time
previewTime	long	The time of the preview referenced in the change request, if any
previewId	long	Numeric preview ID

messageList	complex	Container for messages describing the result of the action
message	complex	Container for a result message
msgDesc	string	Text of the result message, indicating order status, success or failure, additional requirements that must be met before placing the order, etc. Applications typically display this message to the user, which may result in further user action.
msgCode	integer	Standard numeric code of the result message. Refer to the Error Messages documentation for examples. May optionally be displayed to the user, but is primarily intended for internal use.
optionSymbol	complex	Container for the option identifier
symbol	string	The market symbol for the underlier
callOrPut	string	Option type - specifies either CALL or PUT
strikePrice	double	The strike price for this option
expirationYear	integer	The 4-digit year the option will expire
expirationMonth	integer	The month (1-12) the option will expire
expirationDay	integer	The day (1-31) the option will expire
quantity	integer	The number of shares to buy or sell
reserveOrder	boolean	If TRUE, this is a reserve order - meaning that only a limited number of shares will be publicly displayed, instead of the entire order, to avoid influencing other traders.
reserveQuantity	integer	The number of shares to be publicly displayed if this is a reserve order
orderTerm	string	Specifies the term for which the order is in effect. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)
priceType	string	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE
limitPrice	double	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Returned if <code>priceType</code> is LIMIT.
stopPrice	double	The price at which a stock is to be bought or sold if specified in a stop order. Returned if <code>priceType</code> is STOP.
stopLimitPrice	double	The designated price for a stop-limit order. Required if <code>priceType</code> is STOP_LIMIT.

orderAction	string	The action that the broker is requested to perform. Possible values are: • BUY • SELL • BUY_OPEN • SELL_OPEN
routingDestination	string	The exchange where the user requests that the order be executed. Possible values are: • AUTO • AMEX • BOX • CBOE • ISE • NOM • NYSE • PHX

Sample Request

Request URL

POST <https://etwssandbox.etrade.com/order/sandbox/rest/placechangeoptionorder>

Request Parameters - XML

```
<placeChangeOptionOrder xmlns="http://order.etws.etrade.com">
  <changeOptionOrderRequest>
    <accountId>83405188</accountId>
    <orderNum>258</orderNum>
    <clientOrderId></clientOrderId>
    <previewId></previewId>
    <limitPrice>10</limitPrice>
    <stopPrice></stopPrice>
    <stopLimitPrice></stopLimitPrice>
    <allOrNone></allOrNone>
    <quantity>2</quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
    <priceType>LIMIT</priceType>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
  </changeOptionOrderRequest>
</placeChangeOptionOrder>
```

Request Parameters - JSON

```
{
  "placeChangeOptionOrder": {
    "-xmlns": "http://order.etws.etrade.com",
    "changeOptionOrderRequest": {
      "accountId": "83405188",
```

```

        "orderNum": "258",
        "limitPrice": "10",
        "quantity": "2",
        "priceType": "LIMIT",
        "orderTerm": "GOOD_FOR_DAY"
    }
}

```

Sample response - XML

```

<PlaceChangeOptionOrderResponse>
  <optionOrderResponse>
    <accountId>83405188</accountId>
    <allOrNone>false</allOrNone>
    <estimatedCommission>9.49</estimatedCommission>
    <estimatedTotalAmount>2054.53</estimatedTotalAmount>
    <messageList>
      <message>
        <msgDesc>
          Your order was successfully entered during market hours.
        </msgDesc>
        <msgCode>1026</msgCode>
      </message>
    </messageList>
    <orderNum>260</orderNum>
    <orderTime>1269431139424</orderTime>
    <previewTime>0</previewTime>
    <previewId>0</previewId>
    <quantity>2</quantity>
    <reserveOrder>false</reserveOrder>
    <reserveQuantity>0</reserveQuantity>
    <orderTerm>GOOD_FOR_DAY</orderTerm>
    <limitPrice>10</limitPrice>
    <optionSymbol>
      <symbol>IBM</symbol>
      <callOrPut>CALL</callOrPut>
      <strikePrice>175.000000</strikePrice>
      <expirationYear>2010</expirationYear>
      <expirationMonth>10</expirationMonth>
      <expirationDay>16</expirationDay>
    </optionSymbol>
    <orderAction>BUY_OPEN</orderAction>
    <priceType>LIMIT</priceType>
  </optionOrderResponse>
</PlaceChangeOptionOrderResponse>

```

Sample response – JSON

```

{
  "PlaceChangeOptionOrderResponse": {
    "optionOrderResponse": {

```

```

    "accountId": "83405188",
    "allOrNone": "FALSE",
    "estimatedCommission": "9.49",
    "estimatedTotalAmount": "2054.53",
    "messageList": {
      "message": {
        "msgDesc": "Your order was successfully entered during market hours.",
        "msgCode": "1026"
      }
    },
    "orderNum": "260",
    "orderTime": "1269431139424",
    "previewTime": "0",
    "previewId": "0",
    "quantity": "2",
    "reserveOrder": "FALSE",
    "reserveQuantity": "0",
    "orderTerm": "GOOD_FOR_DAY",
    "limitPrice": "10",
    "optionSymbol": {
      "symbol": "IBM",
      "callOrPut": "CALL",
      "strikePrice": "175.000000",
      "expirationYear": "2010",
      "expirationMonth": "10",
      "expirationDay": "16"
    },
    "orderAction": "BUY_OPEN",
    "priceType": "LIMIT"
  }
}

```

Notes

- For equity and option orders, types STOP and STOP LIMIT represent the ["Stop On Quote"](#) and ["Stop Limit on Quote"](#) price types offered by E*TRADE, respectively.
- For equity orders, use limit price for type LIMIT, stop price for type STOP, and both prices for type STOP_LIMIT. For option orders, use limit limit price for type LIMIT, stop price for type STOP, and stop-limit price for type STOP_LIMIT.
- The `clientOrderID` element is used to ensure that a duplicate order is not inadvertently submitted. Within a given account, `clientOrderID` must be unique for every order (regardless of which APIs are used).

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
---------	----------	--------------

Place order	On the order screen, enable the menu option to commit the change only when displaying the response from the preview API. In addition to the change preview (and optionally the original order), show account ID, name, and balances, as well as product fundamentals, intraday, and current (streaming) price info. Check market rules for option orders and display warnings if needed. Display response.	Get Quote, List Accounts, Get Account Balance, Streaming API
Display order status	When placing or changing an order, register for push notifications on the account. Have a listener process on each page appropriately - for instance, be prepared to display order status updates on the completed order display, order list display, or account display.	<i>See Notifications and Streaming API documentation for details.</i>

Related APIs

Preview Option Order Change, Place Option Order, Place Equity Order Change

Sandbox Samples

The following is an example of a request and response in the sandbox environment. Note that the HTTP POST method is used.

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/placechangeoptionorder
```

Request Parameters - XML

```
<placeChangeOptionOrder xmlns="http://order.etws.etrade.com">
  <changeOptionOrderRequest>
    <stopLimitPrice></stopLimitPrice>
    <priceType></priceType>
    <orderTerm>GOOD_UNTIL_CANCEL</orderTerm>
    <accountId>83405188</accountId>
    <orderNum>162</orderNum>
    <clientOrderId>asdf1234</clientOrderId>
    <limitPrice></limitPrice>
    <previewId></previewId>
    <stopPrice></stopPrice>
    <allOrNone></allOrNone>
    <quantity></quantity>
    <reserveOrder></reserveOrder>
    <reserveQuantity></reserveQuantity>
  </changeOptionOrderRequest>
</placeChangeOptionOrder>
```

Response

```
<PlaceChangeOptionOrderResponse>
```



```

<optionOrderResponse>
  <accountId>83405188</accountId>
  <allOrNone>false</allOrNone>
  <estimatedCommission>8</estimatedCommission>
  <estimatedTotalAmount>6308.056</estimatedTotalAmount>
  <messageList>
    <message>
      <msgDesc>Your order was successfully entered during market hours.</msgDesc>
      <msgCode>1026</msgCode>
    </message>
  </messageList>
  <orderNum>1350</orderNum>
  <orderTime>1267556223779</orderTime>
  <quantity>4</quantity>
  <reserveOrder>false</reserveOrder>
  <reserveQuantity>0</reserveQuantity>
  <orderTerm>GOOD_UNTIL_CANCEL</orderTerm>
  <optionSymbol>
    <symbol>MSFT</symbol>
    <callOrPut>CALL</callOrPut>
    <strikePrice>12.500000</strikePrice>
    <expirationYear>2010</expirationYear>
    <expirationMonth>4</expirationMonth>
    <expirationDay>17</expirationDay>
  </optionSymbol>
  <orderAction>BUY_OPEN</orderAction>
  <priceType>MARKET</priceType>
</optionOrderResponse>
</PlaceChangeOptionOrderResponse>

```

Cancel Order

Cancels an order.

Description

This API marks an order as cancelled, if the order has not already been successfully executed. The order continues to appear in responses to the List Orders API for the rest of that day (Eastern time). Note that no order record is deleted.

For timely updates on order status, including cancellation status, use the streaming order status API, detailed elsewhere in this documentation.

URL

<https://etws.etrade.com/order/rest/cancelorder>

HTTP Method: POST

Since this is a POST request, the parameters are included in the request as XML or JSON.

Request Parameters

Parameter	Type	Description
accountId	integer	Numeric account ID
orderNum	integer	Numeric ID for this order

Response Properties

Property	Type	Description
accountId	integer	Numeric account ID
orderNum	integer	Numeric ID for this order
cancelTime	long	The time the cancel request was submitted, in epoch time
resultMessage	string	Result message

Sample Request

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/cancelorder
```

Request Parameters - XML

```
<cancelOrder xmlns="http://order.etws.etrade.com">
  <cancelOrderRequest>
    <accountId>83405188</accountId>
    <orderNum>262</orderNum>
  </cancelOrderRequest>
</cancelOrder>
```

```
</cancelOrderRequest>
</cancelOrder>
```

Request Parameters - JSON

```
{
  "cancelOrder": {
    "-xmlns": "http://order.etws.etrade.com",
    "cancelOrderRequest": {
      "accountId": "83405188",
      "orderNum": "262"
    }
  }
}
```

Sample response - XML

```
<CancelOrderResponse>
  <cancelResponse>
    <accountId>83405188</accountId>
    <orderNum>262</orderNum>
    <cancelTime>1269434848649</cancelTime>
    <resultMessage>Cancel Request Placed Successfully</resultMessage>
  </cancelResponse>
</CancelOrderResponse>
```

Sample response – JSON

```
{
  "CancelOrderResponse": {
    "cancelResponse": {
      "accountId": "83405188",
      "orderNum": "262",
      "cancelTime": "1269434848649",
      "resultMessage": "Cancel Request Placed Successfully"
    }
  }
}
```

Sample use cases

Some possible use-cases and workflows are described below.

Purpose	Workflow	Related APIs
Delete order, List current orders	Use the List Orders API to display orders of any desired status. For open orders, display current market price and an option to cancel the order. Display the account name in the page header.	List Orders, Get Quote, List Accounts

Related APIs

List Orders

Sandbox Samples

The following is an example of a request and response in the sandbox environment. Note that the HTTP POST method is used.

Request URL

```
POST https://etwssandbox.etrade.com/order/sandbox/rest/cancelorder
```

Request XML

```
<cancelOrder xmlns="http://order.etws.etrade.com">
  <cancelOrderRequest>
    <accountId>83550325</accountId>
    <orderNum>262</orderNum>
  </cancelOrderRequest>
</cancelOrder>
```

Response

```
<CancelOrderResponse>
  <cancelResponse>
    <accountId>83550325</accountId>
    <orderNum>262</orderNum>
    <cancelTime>1342492932659</cancelTime>
    <resultMessage>Cancel Request Placed Successfully</resultMessage>
  </cancelResponse>
</CancelOrderResponse>
```

Rate Limits

Get Rate Limits

Returns information on rate limits for the current consumer key.

Description

This API returns the rate limits that are applied to the specified consumer key, broken out by the Accounts, Market, and Order API modules.

The response specifies just the consumption rate, expressed as a number of requests and a time interval. For instance, a typical response might state that 7000 requests are allowed over an interval of 3600 seconds (one hour). It also tells how many requests are still available for the current interval, and the time at which the next interval will begin. Once the available requests for an interval are used up, no more requests are accepted for processing until the next interval begins at the time shown.

Error handling

When a request exceeds the limit, the system returns an HTML error page containing the message: "Number of requests exceeded the rate limit set", with the HTTP status code 400. At that point, depending on program design, a typical error-handler might call the `limit` API to determine whether the per-second or per-hour threshold has been exceeded.

If the response indicates that there are requests remaining for the defined interval, then the error was caused by too many requests per second. In that case, an application can re-issue the request after waiting the required fraction of a second.

However, if there are no requests allowed for the rest of the defined interval, the application will need to wait until a new interval begins (at the time indicated in the response). In this rare situation, if a pending action involves a trade or is otherwise time-sensitive, we suggest advising the user of the anticipated delay and requesting confirmation before proceeding.

URL

<https://etws.etrade.com/statuses/rest/limits>

HTTP Method: GET

Request Parameters

Property	Type	Description
oauth_consumer_key	string	The value used by the consumer to identify itself to the service provider.

oauth_token	string	The request token issued by the service provider.
module	string	The API module for which information is requested. Possible values are: ACCOUNTS, MARKET, ORDER.

Response Parameters

Property	Type	Description
oauth_consumer_key	string	The string used by the consumer to identify itself to the service provider
limitIntervalInSeconds	integer	The interval for which the limit applies, in seconds
requestLimit	integer	The number of requests allowed in the specified interval
requestsRemaining	integer	The remaining number of requests that the system will accept from the user in the current time interval
resetTime	string	The time when the next interval will begin
resetTimeEpochSeconds	integer	The time when the next interval will begin, in epoch time

Sample Request

```
https://etws.etrade.com/statuses/rest/limits?oauth_consumer_key=45e5jf05d7fa1021e2c81d3c7ald2e06Z7VANJzqlp00SZR8HXqhEL1rgDORhsYet5UR1C1tlK&module=accounts&oauth_token=HYNojqv1HVl&#47;wmaEU6UBautoy7m5v3qj5bxZuQeT7s&#61;
```

Sample response - XML

```
<RateLimitStatus>
  <consumerKey>d85df5b910e3cb47b2a4cf26c20229ae</consumerKey>
  <limitIntervalInSeconds>3600</limitIntervalInSeconds>
  <requestsLimit>6500</requestsLimit>
  <requestsRemaining>6500</requestsRemaining>
  <resetTime>02-25-2010 22:16:58 GMT</resetTime>
  <resetTimeEpochSeconds>1267136218</resetTimeEpochSeconds>
</RateLimitStatus>
```

Sample response - JSON

```
{
  "RateLimitStatus": {
    "consumerKey": "d85df5b910e3cb47b2a4cf26c20229ae",
    "limitIntervalInSeconds": "3600",
    "requestsLimit": "6500",
    "requestsRemaining": "6500",
    "resetTime": "02-25-2010 22:16:58 GMT",
    "resetTimeEpochSeconds": "1267136218"
  }
}
```

Notes

- The consumer key and token should be URL-encoded since they might contain invalid characters for a URL, such as a slash or ampersand.

Notifications

Get Notifications

Returns system notification messages relevant to the application.

Description

This API returns a list containing system notification messages of interest to the application developer. Within the response, the messages are grouped into two general types: *global* messages, intended for the entire E*TRADE Developer Platform, and *platform* messages, which are intended only for the developer associated with the consumer key. Within each of those two overall types, messages are further grouped into three priority levels: *info* (low), *notice* (medium), and *alert* (high). Optional request parameters may specify a single message type, or a single message priority level, or both.

These messages are intended for developers, not users, and are very infrequent. They might contain information, for instance, about a new system feature, a planned system upgrade, etc.

URL

<https://etws.etrade.com/notification/rest/getmessagelist>

HTTP Method: GET

Request Parameters

Parameter	Type	Required?	Description
messageType	enum	optional	If specified, requests only the messages intended for this specific user, or only the messages intended for all users. If not specified (default), both types of messages are returned. Possible values are: <ul style="list-style-type: none">• GLOBAL - messages for all developers• PLATFORM - messages for this developer
messageTier	enum	optional	If specified, requests only the messages with the specified message priority level, identified by the name of the container element. Possible values are: <ul style="list-style-type: none">• INFO - low priority (returns <code>infoMessages</code>)• NOTICE - medium priority (returns <code>infoMessages</code>)• ALERT - high priority (returns <code>alertMessages</code>)

Response Properties

Property	Type	Description
globalMessages	complex	Container for messages of <code>messageType</code> GLOBAL.
alertMessages, noticeMessages, infoMessages	complex	These containers contain messages grouped by high priority (alert), medium priority (notice), or low priority (info). Correlates to the <code>messageTier</code> parameter in request.
count	integer	The number of messages in this tier (i.e., the number of alerts, or the number of notices, or the number of info messages).
message	complex	Container for a single message
description	string	The message content
subject	string	The message subject
platformMessages	complex	Container for messages of <code>messageType</code> PLATFORM
consumerKey	string	OAuth consumer key that identifies the developer/application for which these messages are intended
alertMessages, noticeMessages, infoMessages	complex	These containers contain messages grouped by high priority (alert), medium priority (notice), or low priority (info), correlating to the <code>messageTier</code> parameter in the request. By default, all three types of messages are returned.
count	integer	The number of messages in this tier (i.e., the number of alerts, or the number of notices, or the number of info messages)
message	complex	Container for a single message
description	string	The message content
subject	string	The message subject

Sample Request

```
GET https://etws.etrade.com/notification/rest/getmessagelist
```

Sample response - XML

```
<getMessageListResponse xmlns="http://notification.etws.etrade.com/">
  <globalMessages>
    <alertMessages>
      <count>1</count>
    </alertMessages>
    <infoMessages>
      <count>2</count>
      <message>
        <description>This is an info message.</description>
        <subject>Info 1</subject>
      </message>
      <message>
        <description>This is also a test.</description>
        <subject>Info 2</subject>
      </message>
    </infoMessages>
    <noticeMessages>
```

```

    <count>1</count>
    <message>
      <description>This is a notice message.</description>
      <subject>Notice 1</subject>
    </message>
  </noticeMessages>
</globalMessages>
<platformMessages>
  <alertMessages>
    <count>1</count>
    <message>
      <description>Test Verification</description>
      <subject>Hello</subject>
    </message>
  </alertMessages>
  <consumerKey>45e56f05d7fa1021e2c81d3c7ald2e06</consumerKey>
  <infoMessages>
    <count>2</count>
    <message>
      <description>Testing.</description>
      <subject>A test message</subject>
    </message>
    <message>
      <description>This is the body of a message.</description>
      <subject>This is the subject.</subject>
    </message>
  </infoMessages>
  <noticeMessages>
    <count>1</count>
    <message>
      <description>Sample message contents.</description>
      <subject>Sample Message</subject>
    </message>
  </noticeMessages>
</platformMessages>
</getMessageListResponse>

```

Sample response - JSON

```

{
  "getMessageListResponse": {
    "-xmlns": "http://notification.etws.etrade.com/",
    "globalMessages": {
      "alertMessages": { "count": "1" },
      "infoMessages": {
        "count": "2",
        "message": [
          {
            "description": "This is an info message.",
            "subject": "Info 1"
          },
          {
            "description": "This is also a test.",

```

```

        "subject": "Info 2"
    }
]
},
"noticeMessages": {
    "count": "1",
    "message": {
        "description": "This is a notice message.",
        "subject": "Notice 1"
    }
}
},
"platformMessages": {
    "alertMessages": {
        "count": "1",
        "message": {
            "description": "Test Verification",
            "subject": "Hello"
        }
    },
    "consumerKey": "45e56f05d7fa1021e2c81d3c7a1d2e06",
    "infoMessages": {
        "count": "2",
        "message": [
            {
                "description": "Testing.",
                "subject": "A test message"
            },
            {
                "description": "This is the body of a message.",
                "subject": "This is the subject."
            }
        ]
    },
    "noticeMessages": {
        "count": "1",
        "message": {
            "description": "Sample message contents.",
            "subject": "Sample Message"
        }
    }
}
}
}

```

Notes

- The messages returned by this API are intended for developers and not typically exposed to ordinary application users.

Error Handling

Overview

If the system encounters a problem while filling a request, the application may receive an error message in one of three ways. Major errors, such as a down service or an authorization error, cause the server to send an HTTP error page. Errors within an API, such as a syntax problem or bad parameter value, usually just return an `Error` object in the selected data format (XML or JSON), rather than the expected response. And in a very few cases, an individual API may include important operational information in the data response. This is documented in those individual API's. (For example, when previewing or placing an order, the API may return a message that should be displayed for the user.)

A typical approach is to check every response for HTTP errors, then API errors, and branch accordingly. In either case, the expected data response (e.g., `OptionChainResponse` or `GetOrderListResponse`) is not delivered.

Placing orders

The APIs for placing orders return response messages. A typical order workflow involves assembling and validating the order onscreen, then passing it to a preview API and reviewing the response until the order is error-free and the user confirms it for placement. At that point the order is passed to the actual place-order API, and again the response is reviewed for errors before displaying a confirmation that the order was placed.

HTTP Errors

An HTTP error code is returned for most major errors, such as a down service, data not found, or an authorization error. The table of HTTP status messages in this documentation lists the possible errors. In many cases, the HTTP status in the header may be sufficient for recognizing and resolving the problem. If not, the problem can be identified by parsing the HTML that is returned, which will look like this example.

HTTP Status 401 - oauth_problem=token_rejected

type Status report

message `oauth_problem=token_rejected`

description `This request requires HTTP authentication (oauth_problem=token_rejected).`

Apache Tomcat/6.0.26

In the case of a simple situation, such as an expired OAuth token, the application might try to resolve the situation (for instance, by renewing the token). In some other cases, the application may choose to simply display the error message to the user.

API Errors

When an error takes place at the API level, the usual result is an `Error` object in place of the expected data response object. Examples include a badly-formed request or a data error - e.g., an attempt to purchase stock with insufficient funds, or to delete an alert that has already been deleted.

The `Error` object contains an `ErrorCode` and an `ErrorMessage`, as shown below.

```
<Error>
  <ErrorCode>2027</ErrorCode>
  <ErrorMessage>This account is not approved for options trading.</ErrorMessage>
</Error>
```

We recommend that an application simply display the error message to the user. The error code may also optionally be displayed, but it is primarily intended for internal use.

Possible values for these are listed in this documentation in separate tables for each API module - Accounts, Market, Order, and Notifications.

HTTP Status Codes

The following table shows the HTTP status codes that may be returned, and the accompanying messages.

Note that a single status code may be associated with more than one message. In those cases, the status message correlates with a specific resource.

Code	Subcode	Applies to	Message
204	11 or 53	Accounts API's	No data found. (not an error condition)
400		Accounts API's	Invalid [parameter] / Input has some type of problem.
400		Order API's	Errors found.
400	1000 - 1050	Market API's	Invalid [parameter] / Input has some type of problem.
400	53	Market API's	No data found.
401		OAuth	oauth_problem=invalid access token oauth_problem=invalid consumer key oauth_problem=invalid nonce oauth_problem=invalid request token oauth_problem=invalid signature oauth_problem=invalid signature method oauth_problem=oauth parameters absent oauth_problem=oauth_version mismatch oauth_problem=realm mismatch oauth_problem=token_rejected
404	1040	Accounts API's	Invalid alert.
404	1000 or 1050	Accounts API's	No input Account ID's.
413		Order API's	Request Entity Too Large (or data was expected but not found)
500	9999	Accounts API's	The service you requested is not available at this time. Please try again later.

Account Module Error Messages

This table lists error messages returned by APIs in the Accounts service.

Error Code	Error Message
6	Invalid parameters passed. Please verify input values.
11	No Data Found.
53	No Data Found.
600	Alert has been deleted.
1000	Invalid Account Number: Null value passed for account no.
1001	Invalid Account Number: Invalid length.
1002	Invalid Account Number: Invalid format.
1004	Invalid Symbol.
1005	Invalid expiration date.
1006	Invalid Strike Price. Its either missing or has invalid value.
1007	Invalid Call/Put. Its either missing or has invalid value. Allowed values are CALL or PUT.
1008	Invalid Type Code. The code is either missing or has an invalid value. Allowed values are EQ, OPTN, INDX, MF, FI.
1009	Invalid Count. Count value should be between 1 and 25.
1010	Invalid marker. Maximum allowed length is 32 chars.
1011	No positions matching the symbol you have entered can be found.
1012	Specified account does not have any positions.
1013	Invalid User ID: Null value passed for UserId.
1014	Invalid User ID: Invalid length.
1015	Invalid User ID: Invalid format.
1016	Invalid Account no: Negative value passed for account no.
1017	Invalid Alert ID.
1018	Alert cannot be deleted. Please try again later.
1019	Symbol is missing. Please specify symbol along with type code.
1020	Account Number does not exist for user.
1040	Invalid Alert.
9999	The service you requested is not available at this time. Please try again.

In case of an error in the API, the `Error` response is returned in place of a normal data response. It contains an `ErrorCode` and an `ErrorMessage`, as in this example.

```
<Error>
  <ErrorCode>2027</ErrorCode>
  <ErrorMessage>This account is not approved for options trading.</ErrorMessage>
</Error>
```

Market Module Error Messages

This table lists error messages returned by APIs in the Market service.

Error Code	Error Message
53	No Data Found.
1002	Invalid ReqId Format.
1003	Invalid Expiration Date.
1004	Invalid Spread Date.
1005	Invalid Price Type.
1006	Invalid Strike Price.
1008	Invalid Symbol.
1010	Invalid UserId.
1011	Invalid ReqId Format.
1012	Invalid Option Type.
1013	Invalid Security Type.
1014	The request has no symbols.
1015	A request may contain up to 25 quotes.
1016	The user has not signed the real time quote agreement.
1017	The symbol specified is Invalid.
1018	The company or symbol specified is Invalid.
1019	The input data is Invalid. Please verify.
1020	Invalid company name.
1021	Invalid add adjusted flag.
9999	The service you requested is not available at this time. Please try again.

In case of an error in the API, the `Error` response is returned in place of a normal data response. It contains an `ErrorCode` and an `ErrorMessage`, as in this example.

```
<Error>
  <ErrorCode>2027</ErrorCode>
  <ErrorMessage>This account is not approved for options trading.</ErrorMessage>
</Error>
```


Order Module Error Messages

This table lists error messages returned by APIs in the Order service.

Error Code	Error Message
100	Only GTC and Day orders are allowed with All or None.
200	All Or None, Quantity should be greater than equal to 300.
300	Only Limit and Stop Limit on Quote orders are allowed with All or None.
400	Select either Reserve Order and All or None.
500	The Quantity for Reserve Order should be greater than or equal to 1000.
520	Invalid Routing Destination Specified.
530	Invalid AON/Order Term combination.
600	For Reserve Order the Order should be a LIMIT Order.
700	Invalid Reserve Quantity.
800	Invalid Client OrderId.
900	Invalid Preview Id.
1003	This does not appear to be a valid stock symbol. Please make sure that you have entered the symbol correctly.
1005	This symbol is not recognized. Please make sure that you have entered the symbol correctly.
1010	There is currently a 500 share or 10% of volume maximum share quantity allowed in your account. Please adjust the share quantity you entered, or contact Customer Service at 1-800-ETRADE-1 (1-800-387-2331) for assistance.
1026	Your order was successfully entered during market hours.
1028	You have an existing open order for this security on the same side of the market.
1039	You have entered a number that exceeds the number of shares you hold in the security you want to trade. Please go back and enter a number either equal to or less than the number of shares you hold.
1041	Your sell short order cannot be processed. Either you do not have a margin account or short sales in this security are not allowed.
1042	You have an existing open order for this security on the same side of the market. If you did not intend to place a second order for this security, please modify your order now.
1044	Your sell short order cannot be processed. Short sales in this security are not allowed, as we were unable to borrow the shares.
1046	The security for which you have placed an order is on the E*TRADE Securities restricted list.
1051	E*TRADE allows only limit buy orders on Bulletin Board securities. Please go back and enter a valid limit price.
1068	Because the market is now closed, this order will not be reviewed until the morning prior to market open on next regular trading day.
1100	Null value specified.
1101	Invalid format.
1102	Invalid account specified for the user.

1103	Invalid count specified. Count should be between 0 and 25.
1104	Invalid markers. Please provide either beginMarker or endMarker not both.
1105	Invalid Account no: Negative value passed for account no.
1106	Invalid User Id: Null value passed for UserId.
1107	Invalid User Id: Invalid format.
2019	The symbol you entered does not appear to be valid. Please make sure you've entered the symbol correctly.
2027	This account is not approved for options trading.
2078	Orders entered via the Web site must be for whole shares. Your order has been rounded down to the nearest whole number.
2086	This security is not eligible for new advanced orders at this time.
2093	This symbol does not appear to be a stock symbol. Please make sure that you have entered the valid symbol correctly.
2097	Important! NYSE Rule 431 Notice: Because you are currently subject to a day trading minimum equity call, your account will be restricted to cash only transactions for 90 days if this day trade executes.
2098	This order to purchase securities, if accepted and executed, will be paid for out of proceeds from the previous sale of securities in your account, which is to settle on or before the date by which payment for this purchase transaction is due. If you subsequently enter an order to sell these securities before they are fully paid for, your account may be subject to a 90-day restriction under federal securities regulations.
2099	Because these securities were purchased with the proceeds of the sale of another security in your account, the order to sell these securities, if executed prior to the settlement date of the previous sale, may subject your account to a 90-day restriction under federal securities regulations.
3002	"This account is not approved for Level 2 or 3 options trading.
3003	"Please note that this option contract does not equal 100 shares per contract.
3006	Naked index options are not permitted at E*TRADE Securities LLC.
3007	You are not approved for naked calls.
3008	This account is not approved for Level 3 options trading.
3011	We cannot accept this order until your sell order for underlying stock is canceled at your current option trading level.
3015	This account is not approved for Level 3 options trading.
3023	"The ratio of contracts for this order is not one-to-one.
3024	The ratio of shares to contracts for this order is not one-to-one.
7046	You may not place a Sell Short, Market on Close order.
7508	This order cannot be accepted because a Reserve Order can be routed to ECNs only. Please select ARCA, INET, or NSDQ for the market center.
36103	The ratio of shares to contracts for this order is not one-to-one.
36104	No Records Found.
36105	The ratio of contracts for this order is not one-to-one.
39999	For sandbox testing - order number is inconsistent with security type.
40000	Order number is either missing or invalid. Please specify valid order number.

41000	This order cannot be changed using this Application, it can only be cancelled. To change please log into the E*TRADE website.
600000	Please specify Option Type (Call/Put).
610000	Invalid strike price specified.
620000	Invalid expiration year specified.
630000	Sent too many requests at the same time.
640000	Invalid expiration date. Please check the input values for expirationYear, expirationMonth and expirationDay.
650000	Invalid All Or None specified.
660000	Invalid Reserve order specified.

In case of an error in the API, the `Error` response is returned in place of a normal data response. It contains an `ErrorCode` and an `ErrorMessage`, as in this example.

```
<Error>
  <ErrorCode>2027</ErrorCode>
  <ErrorMessage>This account is not approved for options trading.</ErrorMessage>
</Error>
```

Notification Error Messages

This table lists error messages returned by the Notification service.

Error Code	Error Message
3000	Failed to get messages.
4000	Invalid Consumer Key.
5000	Enter the subject of the message.
6000	Enter the message description.
7000	Message Id is missing or invalid Message ID entered.
8000	Message Tier is missing or invalid Message Tier entered.
9000	Message Type is missing or invalid Message Type entered.
10000	Create By is missing.
11000	Update By is missing.
12000	Status is missing or invalid Status entered.
13000	Global messages cannot have same subject.
14000	Platform messages cannot have same subject.
15000	No Data Found for given Message ID.

In case of an error in the API, the `Error` response is returned in place of a normal data response. It contains an `ErrorCode` and an `ErrorMessage`, as in this example.

```
<Error>
  <ErrorCode>2027</ErrorCode>
  <ErrorMessage>This account is not approved for options trading.</ErrorMessage>
</Error>
```

Code Resources

The E*TRADE Developer Platform includes a number of code resources:

- [A tutorial](#) that walks through authorization and a simple transaction using the Java SDK
- Guides to installing and using the SDKs:
 - [Java](#)
 - [PHP](#)
 - [VC++](#)
- Code snippets that use the Java and PHP SDKs to perform common trading functions, organized into four modules:
 - [Authorization](#)
 - [Accounts](#)
 - [Market](#)
 - [Orders](#)

To use these resources you'll need to install the appropriate SDKs.

Tutorial: Start Developing in Java

Introduction

This tutorial uses the E*TRADE Java SDK, and is designed for Java programmers who want to develop their own application using the E*TRADE API to leverage E*TRADE's market data offerings, order routing capabilities, and other services.

If you are not a Java developer, you may still find this tutorial helpful - the examples are simple and the code is easy to read. You may also find helpful information elsewhere in the platform documentation. The OAuth authorization process, in particular, is explained in the separate Authorization section of this documentation.

The tutorial will walk you through two procedures:

1. Authorize your application using OAuth.
2. Retrieve a list of your E*TRADE accounts.

Requirements

Java SDK

To proceed with this tutorial, you must first have completed the installation of the E*TRADE Java SDK, including:

- Java 1.6 or later installed
- 3rd-party jars installed
- E*TRADE Java SDK libraries in your CLASSPATH

Sandbox consumer key

The application created in this tutorial will need a valid consumer key for the sandbox test environment, as described in the Introduction documentation.

Step 1: OAuth Authorization

The OAuth protocol enables a user to authorize an application to use a data service on the user's behalf. The process is explained in detail in our Authorization. Note that, in the instructions below, the "user" is generally assumed to be the developer who is reading or following the tutorial.

In OAuth, an application's access to the user account requires a temporary access token and access secret. The token must accompany every API call, along with a signature based on the secret. Your application must perform the following steps to get the access token and secret:

1. Obtain a request token and token secret from E*TRADE, using your consumer key.
2. Redirect the user to an E*TRADE authorization page, where the user logs in and grants access to the application. This results in a verification code, which is then typed into the application by the user or automatically passed to the application via a callback.
3. Use the request token and verification code to get the access token and token secret.

Obtain a request token

The following Java code obtains the request token and token secret. Remember that the request token is only valid for five minutes.

```
import com.etrade.etws.account.Account;
import com.etrade.etws.account.AccountListResponse;
import com.etrade.etws.oauth.sdk.client.IOAuthClient;
import com.etrade.etws.oauth.sdk.client.OAuthClientImpl;
import com.etrade.etws.oauth.sdk.common.Token;
import com.etrade.etws.sdk.client.ClientRequest;

// Variables
public IOAuthClient client = null;
public ClientRequest request = null;
public Token token = null;
public String oauth_consumer_key = null; // Your consumer key
public String oauth_consumer_secret = null; // Your consumer secret
public String oauth_request_token = null; // Request token
public String oauth_request_token_secret = null; // Request token secret

client = OAuthClientImpl.getInstance(); // Instantiate IOAuthClient
request = new ClientRequest(); // Instantiate ClientRequest
request.setEnv(Environment.SANDBOX); // Use sandbox environment

request.setConsumerKey(oauth_consumer_key); //Set consumer key
request.setConsumerSecret(oauth_consumer_secret); // Set consumer secret
token = client.getRequestToken(request); // Get request-token object
oauth_request_token = token.getToken(); // Get token string
oauth_request_token_secret = token.getSecret(); // Get token secret
```

Obtain a verification code

The next section of code redirects your user to E*TRADE for authorization. Upon successful login, the E*TRADE page displays information to user about your application's request for access. If the user approves, the page displays a verification code.

```
import java.awt.Desktop;
import java.net.URI;

String authorizeURL = null;
authorizeURL = client.getAuthorizeUrl(request); // E*TRADE authorization URL
URI uri = new java.net.URI(authorizeURL);
```

```
Desktop desktop = Desktop.getDesktop();  
desktop.browse(authorizeURL);
```

If Customer Service has configured a callback URL for your application, the user is then automatically routed to your callback URL, with the verification code added as a parameter on the URL. If not, the user has to manually copy the verification code and enter it into your application. More information on callbacks is provided in our Authorization documentation.

Obtain an access token

The following code shows how to exchange the request token and verification code for an access token with the verification code.

```
public String oauth_access_token = null; // Variable to store access token  
public String oauth_access_token_secret= null; // Variable to store access token  
secret  
public String oauth_verify_code = "Your verification_code"; // Should contain the  
Verification Code received from the authorization step  
  
request = new ClientRequest(); // Instantiate ClientRequest  
request.setEnv(Environment.SANDBOX); // Use sandbox environment  
  
// Prepare request  
request.setConsumerKey(oauth_consumer_key); // Set consumer key  
request.setConsumerSecret(oauth_consumer_secret); // Set consumer secret  
request.setToken(oauth_request_token); // Set request token  
request.setTokenSecret(oauth_request_token_secret); // Set request-token secret  
request.setVerifierCode(oauth_verify_code); // Set verification code  
  
// Get access token  
token = client.getAccessToken(request); // Get access-token object  
oauth_access_token = token.getToken(); // Access token string  
oauth_access_token_secret = token.getSecret(); // Access token secret
```

Congratulations, your application has been authorized - at least, temporarily. A sandbox access token is good for 24 hours. Production access tokens are good until midnight Eastern time, or for a duration configured at your request by Customer Service.

Step 2: Retrieve a list of accounts

You have the access token, so now you can make requests on the user's account. The code below shows how to request a list of the user's accounts and display the results.

```
request = new ClientRequest(); // Instantiate ClientRequest  
  
// Prepare request  
request.setEnv(Environment.SANDBOX);  
request.setConsumerKey(oauth_consumer_key);  
request.setConsumerSecret(oauth_consumer_secret);  
request.setToken(oauth_access_token);
```



```

request.setTokenSecret(oauth_access_token_secret);

try
{
    AccountsClient account_client = new AccountsClient(request);
    AccountListResponse response = account_client.getAccountList();
    List<Account> alist = response.getResponse();
    Iterator<Account> al = alist.iterator();
    while (al.hasNext()) {
        Account a = al.next();
        System.out.println("=====");
        System.out.println("Account: " + a.getAccountId());
        System.out.println("=====");
    }
}
catch (Exception e)
{
}

```

What's next?

Now that you've authorized your application and retrieved an account list, you can experiment with other E*TRADE APIs on your own. We encourage you to explore these other resources:

- [REST API reference](#)
- [SDK documentation and code snippets](#)

E*TRADE SDK Guides

Using the Java SDK

Introduction

The E*TRADE Java SDK is a library of Java methods written on top of the E*TRADE REST API. It provides Java developers with convenient, Java-based access to our trading platform.

*What you can do with the E*TRADE Java SDK*

The Java SDK allows you to programmatically execute trades directly through the E*TRADE REST API in your Java application, without having to deal with the mechanics of HTTP. It provides code libraries for the following functional areas:

- Authentication
- Accounts
- Market Data
- Orders

How to install the E*TRADE Java SDK

The E*TRADE Java SDK requires JDK 1.6.

In addition, it requires that the following 3rd-party jars be present and included in your CLASSPATH. You can download them from the listed sites if necessary.

3rd-party libraries	Links for downloading them if necessary
log4j-1.2.15.jar	http://www.apache.org/dyn/closer.cgi/logging/log4j/1.2.15/apache-log4j-1.2.15.tar.gz
commons-codec-1.3.jar	http://commons.apache.org/codecs/
commons-httpclient-3.1.jar	http://hc.apache.org/downloads.cgi
xstream-1.3.1.jar	http://xstream.codehaus.org/download.html
commons-lang-2.4.jar	http://commons.apache.org/lang/download_lang.cgi
commons-logging-1.0.4.jar	http://commons.apache.org/logging/download_logging.cgi

Finally, you'll need the SDK itself, which can be downloaded from the E*TRADE developer website at this URL:

https://content.etrade.com/etrade/extras/All_Jar.zip

The All_Jar.zip file expands into the following libraries:

Filename	Library
etws-common-connections-1.0.jar	Dependent library
etws-oauth-sdk-1.0.jar	OAuth
etws-accounts-sdk-1.0.jar	Accounts
etws-market-sdk-1.0.jar	Market Data
etws-order-sdk-1.0.jar	Orders

You can begin using the SDK as soon as it's installed. For details on data structures, data types, options, and other details of individual API features, consult the REST API reference documentation.

E*TRADE Java Methods

This table lists the methods provided by the E*TRADE Java SDK.

Module	Method	Description
OAuth	getRequestToken	Returns Request Token
	getAuthorizeURL	Returns URL for E*TRADE authorization page
	getAccessToken	Returns access token
	renewAccessToken	Renews an access token
	revokeAccessToken	Revokes an access token
Accounts	getAccountList	Returns the complete list of E*TRADE accounts for the current user, including IRA, savings, checking, etc.
	getAccountBalance	Returns the current account balance and related details for a specified account
	getAccountPositions	Returns the positions held in the specified account
	getAlerts	Lists alerts for the current user
	getAlertDetails	Retrieves alert detail
	deleteAlert	Method to delete a particular alert
Market	getQuote	Returns current market information for equities and options
	productLookup	Looks up the exchange and symbol of a security based on the name
	getExpiryDates	Returns a list of the expiration dates for options that have the specified underlier
	getOptionChain	Returns a list of option chains associated with a specific underlier
Order	getOrderList	Returns a list of open orders or orders updated during the current day
	previewEquityOrder	Returns a preview of an equity order
	placeEquityOrder	Submits an equity order
	previewChangeEquityOrder	Returns a preview of a change to an equity order
	placeChangeEquityOrder	Executes a change equity order request.
	previewOptionOrder	Returns a preview of an option order

	placeOptionOrder	Submits an option order
	previewChangeOptionOrder	Returns a preview of a change to an option order
	placeChangeOptionOrder	Executes a change to an option order
	cancelOrder	Cancels an order

Using the PHP SDK

Introduction

This SDK provides an easy-to-use client for the E*TRADE REST API. It gives PHP developers a convenient, PHP-friendly way to take advantage of our REST API and the E*TRADE Developer Platform.

What you can do with E*TRADE PHP SDK

The PHP SDK allows you to programmatically execute trades directly through the E*TRADE REST API in your PHP application. It provides libraries for the following functional areas:

- Authentication
- Accounts
- Market Data
- Orders

Requirements

Our PHP SDK requires PHP version 5.2 or newer. You can download the latest version of PHP for your operating system at <http://www.php.net>. Your version of PHP must include curl lib support.

Additionally, our SDK requires Google's OAuth library for PHP. After you've installed the PHP SDK, download the OAuth library from <http://oauth.googlecode.com/svn/code/php/OAuth.php> (Subversion revision: 1263), and copy it into the SDK's OAuth folder.

Installation

1. Download the PHP-SDK from the E*TRADE website to a local folder of your choice.
2. Create a `config.php` file for your application. A sample `config.php` can be copied from the SDK's `Samples` directory.
3. Include your local `config.php` at the top of your application.
4. Include `<SDK_PATH>/Common/common.php` immediately after that.

Once this is done, you can start using the SDK.

For details on data structures, data types, and other API details, consult the REST API reference documentation.

Using the Config file

The E*TRADE PHP SDK provides a default configuration file in the `Common` folder as `config.php`. This file provides the basic configurations required by the SDK. You can override any or all configuration values by creating your own `config.php` file under your application directory and including it at the start of your application.

Some key configuration settings are:

Config setting	Description
ET_SDK_PATH	Location of your PHP-SDK directory
DEBUG_MODE	Use this setting to provide debug messages during development
RESPONSE_FORMAT	Selects the format of responses returned by the API. Possible values are "xml" and "json".

The remaining configuration values provide URL's for using APIs. Please read `config.php` for details.

Libraries

The E*TRADE PHP SDK includes the following libraries to let you interface with the E*TRADE Developer Platform in PHP.

Library	Description
etOAUTH	Provides all functions needed for OAuth authorization
etAccounts	Provides an interface to the E*TRADE Accounts API
MarketClient	Provides an interface to the E*TRADE Market API
OrderClient	Provides an interface to the E*TRADE Orders API

E*TRADE PHP Methods

OAuth

The E*TRADE Developer Platform uses the OAuth ([1.0a](#)) open authentication protocol. This library provides all the necessary OAuth functionality.

Class	Constructor / Method	Description
etOAuthConsumer		Provides a consumer object to use for OAuth operations
	__construct()	Params: API key, API secret, callback method (optional; default is "oob") Returns: etOAuthConsumer object
etOAuth		The etOAuth class provides the following methods to complete OAuth operations:

	<code>__construct()</code>	This is the class constructor; it constructs a request object using the consumer object, e.g.: <pre>\$consumer = new etOAuthConsumer('KEY','SECRET'); \$request = new etOAuth(\$consumer);</pre>
	<code>GetRequestToken()</code>	Params: none Returns: Array of request token values.
	<code>GetAuthorizeURL()</code>	Params: none Returns: String, authorize URL
	<code>GetAccessToken()</code>	Params: verifier code. Returns: Array, access token
	<code>RenewAccessToken()</code>	Params: none Returns: xml message returned from API
	<code>RevokeAccessToken ()</code>	Params: none Returns: xml message returned from API

Accounts

The Accounts library, "etAccounts", provides methods to access account-related resources. To call Accounts methods, you must have an access token.

Class	Constructor / Method	Description
etAccounts	<code>__construct()</code>	Params: consumer object Returns: etAccounts object
	<code>GetAccountList()</code>	Provides list of accounts for the user Params: none Returns: list of accounts as XML or JSON
	<code>GetAccountBalance()</code>	Provides account balance details Params: account ID Returns: account balance as XML or JSON

	GetAccountPositions()	<p>Returns a list of account positions</p> <p>Params: account ID, AccountPositionsRequest object</p> <p>AccountPositionsRequest object properties are:</p> <ul style="list-style-type: none"> • count • marker • symbol • typeCode • callPut • expYear • expMonth • expDay • strikePrice <p>Returns: account positions as XML or JSON</p> <p>For data type, options, and other details of each parameter, refer to the REST API reference.</p>
	GetAlerts()	<p>Provides list of alerts</p> <p>Params: none</p> <p>Returns: List of alerts as XML or JSON</p>
	GetAlertDetails()	<p>Provides alert details</p> <p>Params: alert ID</p> <p>Returns: Details of an alert as XML or JSON</p>
	DeleteAlert()	<p>Deletes an alert</p> <p>Params: alert ID</p> <p>Returns: string containing success or failure message from server</p>

Market Data

The Market library, "MarketClient", provides an interface to the ETRADE Market API.

Class	Constructor / Method	Description
MarketClient	__construct()	<p>Creates an object of MarketClient class, using consumer object</p> <p>Params: consumer object</p> <p>Returns: market client object</p>
	getOptionChain()	<p>Returns an option chain</p> <p>Params: getOptionChainsParams object</p> <p>Valid getOptionChainsParams properties:</p> <ul style="list-style-type: none"> • underlier • chainType • skipAdjusted • expirationMonth • expirationYear <p>Returns: option chains as XML or JSON</p>

	productLookup()	Returns product details. Params: productLookupParams object Valid productLookupParams object properties • company • type Returns: product detail as XML or JSON
	getExpiryDates()	Returns expiry dates of a product. Params: getExpiryDateParams object. Valid getExpiryDateParams properties: • underlier • expiryType Returns: expiry date list as XML or JSON
	getQuote()	Returns a quote for a product. Params: getExpiryDateParams object Valid getExpiryDateParams properties: • symbolList • afterhourFlag • detailFlag Returns: quote detail as XML or JSON

Orders

The Orders library, "OrderClient", provides an interface for the ETRADE Orders API.

Class	Constructor / Method	Description
OrderClient	getOrderList()	Provides a list of orders that are open or were active during the current day Params: Account ID Returns: Orders list as XML or JSON

	previewEquityOrder()	<p>Provides preview of an equity order. Params: getExpiryDateParams object Valid getExpiryDateParams properties:</p> <ul style="list-style-type: none"> • accountId • clientId • limitPrice • previewId • stopPrice • allOrNone • quantity • reserveOrder • reserveQuantity • stopLimitPrice • symbol • orderAction • priceType • routingDestination • marketSession • orderTerm <p>Returns: response as XML or JSON</p>
	placeEquityOrder()	<p>Places an equity order. Params: getExpiryDateParams object Valid getExpiryDateParams properties:</p> <ul style="list-style-type: none"> • accountId • clientId • limitPrice • previewId • stopPrice • allOrNone • quantity • reserveOrder • reserveQuantity • stopLimitPrice • symbol • orderAction • priceType • routingDestination • marketSession • orderTerm <p>Returns: response as XML or JSON</p>

	previewOptionOrder()	<p>Provides preview of an option order. Params: OptionOrderRequest object. Valid OptionOrderRequest properties:</p> <ul style="list-style-type: none"> • accountId • clientId • limitPrice • previewId • stopPrice • allOrNone • quantity • reserveOrder • reserveQuantity • stopLimitPrice • symbolInfo (object) • orderAction • priceType • routingDestination • marketSession • orderTerm <p>Valid symbolInfo object properties:</p> <ul style="list-style-type: none"> • symbol • callOrPut • strikePrice • expirationYear • expirationMonth • expirationDay <p>Returns: response as XML or JSON</p>
--	----------------------	--

	placeOptionOrder()	<p>Returns expiry dates of a product. Params: OptionOrderRequest object. Valid OptionOrderRequest properties:</p> <ul style="list-style-type: none"> • accountId • clientId • limitPrice • previewId • stopPrice • allOrNone • quantity • reserveOrder • reserveQuantity • stopLimitPrice • symbolInfo (object) • orderAction • priceType • routingDestination • marketSession • orderTerm <p>Valid symbolInfo object properties:</p> <ul style="list-style-type: none"> • symbol • callOrPut • strikePrice • expirationYear • expirationMonth • expirationDay <p>Returns: response as XML or JSON</p>
	previewChangeEquityOrder()	<p>Provides preview of change to an equity order Params: changeEquityOrderRequest object. Valid changeEquityOrderRequest properties:</p> <ul style="list-style-type: none"> • priceType • orderTerm • accountId • orderNum • clientId • limitPrice • previewId • stopPrice • allOrNone • quantity • reserveOrder • reserveQuantity <p>Returns: response as XML or JSON</p>

	<code>previewChangeOptionOrder()</code>	<p>Provides preview of change to an option order. Params: <code>changeOptionOrderRequest</code> object Valid <code>changeOptionOrderRequest</code> properties:</p> <ul style="list-style-type: none"> • <code>stopLimitPrice</code> • <code>priceType</code> • <code>orderTerm</code> • <code>accountId</code> • <code>orderNum</code> • <code>clientOrderId</code> • <code>limitPrice</code> • <code>previewId</code> • <code>stopPrice</code> • <code>allOrNone</code> • <code>quantity</code> • <code>reserveOrder</code> • <code>reserveQuantity</code> <p>Returns: response as XML or JSON</p>
	<code>placeChangeOptionOrder()</code>	<p>Executes change to an option order. Params: <code>changeOptionOrderRequest</code> object Valid <code>changeOptionOrderRequest</code> properties:</p> <ul style="list-style-type: none"> • <code>stopLimitPrice</code> • <code>priceType</code> • <code>orderTerm</code> • <code>accountId</code> • <code>orderNum</code> • <code>clientOrderId</code> • <code>limitPrice</code> • <code>previewId</code> • <code>stopPrice</code> • <code>allOrNone</code> • <code>quantity</code> • <code>reserveOrder</code> • <code>reserveQuantity</code> <p>Returns: response as XML or JSON</p>
	<code>cancelOrder()</code>	<p>Cancels an order. Params: <code>CancelOrderRequest</code> object Valid <code>CancelOrderRequest</code> properties:</p> <ul style="list-style-type: none"> • <code>accountId</code> • <code>orderNum</code> <p>Returns: response as XML or JSON</p>

Using the VC++ SDK

Introduction

This SDK provides an easy-to-use client for the E*TRADE REST API, offering VC++ developers convenient access to our trading platform.

*What you can do with E*TRADE VC++ SDK*

The VC++ SDK allows you to programmatically execute trades directly through the E*TRADE REST API in your VC++ application. It provides libraries for the following functional areas:

- Authentication
- Accounts
- Market Data
- Orders

Requirements

Our VC++ SDK requires the Visual C++ 2010 Redistributable package from Microsoft and the Code Synthesis XSD compiler and libraries from Code Synthesis.

You can download the VC++ 2010 Redistributable package at: <http://www.microsoft.com/en-us/download/details.aspx?id=5555>.

You can download the XSD 3.3 installer at:
<http://www.codesynthesis.com/products/xsd/download.xhtml>.

Installation

1. Create a directory, e.g., "C:\VC++ SDK".
2. Download the VC++ SDK zip file from our developer website.
3. Extract the files into the directory you created.

Once this is done, you can start using the SDK.

For details on data structures, data types, options, and other details of individual API features, consult the E*TRADE REST API Reference.

Libraries

The VC++ SDK includes the following libraries for use with the E*TRADE Developer Platform.

Library	Description
ETCommon.dll, ETCommon.lib	Common resources required by all other libraries

ETOAuth.dll, ETOAuth.lib	OAuth authorization library
ETAccount.dll, ETAccount.lib	Access to account-related services, transaction history, and alerts.
ETMarket.dll, ETMarket.lib	Interface to the E*TRADE Market API
ETOrder.dll, ETOrder.lib	Interface to the E*TRADE Orders API

Authorization

The E*TRADE Developer Platform uses the [OAuth 1.0a](#) open authentication protocol. This library provides the necessary authorization functionality. Please refer to the guide on [Authorizing the Application](#) for important information on using OAuth with the E*TRADE Developer Platform.

Get Request Token

```
bool COAuthSDK::GetRequestToken(CClientDetails &objClientDetails)
```

This API acquires an OAuth request token using the consumer key and consumer secret. It throws exception if `m_strConsumerKey` or `m_strConsumerSecret` parameter(s) is empty.

CClientDetails

Name	IN/OUT	Description
m_environment	IN	Optional. Possible values are SANDBOX (default) and LIVE.
m_strConsumerKey	IN	OAuth consumer key provided by E*TRADE
m_strConsumerSecret	IN	OAuth consumer secret provided by E*TRADE
m_strToken	OUT	Returned by the function if successful
m_strTokenSecret	OUT	Returned by the function if successful
m_strCallback	IN	Optional; default value is "oob"

Authorize Application

```
string COAuthSDK::AuthorizeUrl(CClientDetails &objClientDetails)
```

This API redirects the user to the E*TRADE authorization site to log in and receive a verification code.

CClientDetails

Name	IN/OUT	Description
m_environment	IN	Optional. Possible values are SANDBOX (default) and LIVE.
m_strConsumerKey	IN	Consumer key provided by E*TRADE
m_strConsumerSecret	IN	Optional

m_strToken	IN/OUT	This parameter serves a dual purpose when called as IN parameter. Function sets this parameter to empty string if it is successful.
m_strTokenSecret	IN/OUT	Optional
m_strCallback	IN	Optional; default value is "oob".

Get Access Token

```
bool COAuthSDK::GetAccessToken(CClientDetails &objClientDetails,string strVerifier)
```

This API is used to get an OAuth access token using consumer key, consumer secret, request token, and request secret. This function throws exception if m_strConsumerKey, m_strConsumerSecret, m_strToken, or m_strTokenSecret is empty.

CClientDetails

Name	IN/OUT	Description
m_environment	IN	Optional. Possible values are SANDBOX (default) and LIVE.
m_strConsumerKey	IN	Consumer key
m_strConsumerSecret	IN	Consumer secret
m_strToken	IN/OUT	This parameter serves a dual purpose. It accepts the request token as input and returns the access token.
m_strTokenSecret	IN/OUT	This parameter serves a dual purpose. It takes the request token secret as input and returns the access token secret.
m_strCallback	IN	Optional; default value is "oob".
strVerifier	IN	The verification code received by the user after authenticating with the E*TRADE platform

Access Protected Resources

```
string COAuthSDK::GetProtectedResource(CClientDetails &objClientDetails,string strUrl, HttpMethodConstants httpMethod , string postParameters)
```

This API can call any REST E*TRADE API. Function will return XML by default; for JSON response, add .json to the URL.

CClientDetails

Name	IN/OUT	Description
m_environment	IN	Optional. Possible values are SANDBOX (default) and LIVE.
m_strConsumerKey	IN	Consumer key
m_strConsumerSecret	IN	Consumer secret
m_strToken	IN/OUT	This parameter set by GetAccessToken

m_strTokenSecret	IN/OUT	This parameter set by GetAccessToken
m_strCallback	IN	Optional; default value is "oob".
strUrl	IN	This is the URL of the desired REST API, as described in the REST API reference documentation. For example: https://etws.etrade.com/accounts/rest/accountpositions/12345678?count=10
httpMethod	IN	The HTTP method to use. Possible values are: <ul style="list-style-type: none"> GETMethod (default) POSTMethod DELETEMETHOD
postParameters	IN	<p>This parameter contains XML for the HTTP request body. It is required only if using the HTTP POST method. For example:</p> <pre><PlaceEquityOrder xmlns="http://order.etws.etrade.com"> <EquityOrderRequest> <accountId>12345678</accountId> <clientId> test23 </clientId> <limitPrice>19</limitPrice> <quantity>90</quantity> <symbol>ETFC</symbol> <orderAction>BUY</orderAction> <priceType>LIMIT</priceType> <marketSession> REGULAR </marketSession> <orderTerm>GOOD_FOR_DAY</orderTerm> </EquityOrderRequest> </PlaceEquityOrder></pre>

Renew Access Token

```
void COAuthSDK:: RenewToken(CClientDetails &objClientDetails)
```

This API is used to renew an Access Token. If successful, it returns "Access Token has been renewed".

CClientDetails

Name	IN/OUT	Description
m_environment	IN	Optional. Possible values are SANDBOX (default) and LIVE.
m_strConsumerKey	IN	Consumer key
m_strConsumerSecret	IN	Consumer secret
m_strToken	IN/OUT	This parameter serves a dual purpose. It accepts the request token as input and returns the access token.

m_strTokenSecret	IN/OUT	This parameter serves a dual purpose. It takes the request token secret as input and returns the access token secret.
m_strCallback	IN	Optional; default value is "oob".

Revoke Access Token

```
void COAuthSDK::RevokeToken(CClientDetails &objClientDetails) throw
(...)
```

This API is used to revoke an existing access token. If successful, it returns "Revoked Access Token", and the access token becomes unusable.

CClientDetails

Name	IN/OUT	Description
m_environment	IN	Optional. Possible values are SANDBOX (default) and LIVE.
m_strConsumerKey	IN	Consumer key
m_strConsumerSecret	IN	Consumer secret
m_strToken	IN/OUT	This parameter serves a dual purpose. It accepts the request token as input and returns the access token.
m_strTokenSecret	IN/OUT	This parameter serves a dual purpose. It takes the request token secret as input and returns the access token secret.
m_strCallback	IN	Optional; default value is "oob".

Accounts

The Accounts library provides methods to access account-related resources.

List Accounts

```
AccountList CAccountSDK:: GetAccountList()
```

This API returns a list of accounts associated with the user. It takes no parameters.

Get Account Balance

```
CAccountBalanceResponse CAccountSDK:: GetAccountBalance(CString
strAccountId)
```

This API returns the current account balance details for a specified account.

Type	Name	Description
------	------	-------------

CString	strAccountId	Numeric account ID
---------	--------------	--------------------

Get Account Positions

```
CAccountPositionsResponse CAccountSDK:: GetAccountPositions
(CAccountPositionsRequest accountPositionsRequest)
```

This API returns positions held in the account.

CAccountPositionsRequest

Type	Name	Description
CString	m_AccountId	Numeric account ID
CString	m_Count	Optional. The number of positions in the account. Default is 25.
CString	m_Marker	Additional positions in the account if there are more than 25 positions. This is optional parameter.
CString	m_Symbol	The market trading symbol for the stock being bought or sold.
CString	m_TypeCode	Type of derivative that the position reflects. There are two options for the derivative. <ul style="list-style-type: none"> • EQ (equity) • OPTN (option) • INDX (index) • MF (mutual fund) • FI (fixed income)
CString	m_CallPut	Option parameter that allows the owner the right to either buy or sell an option. Possible values are: CALL, PUT. Required if type code is OPTN.
double	m_StrikePrice	The price at which an option can be exercised. Required if type code is OPTN.
CString	m_ExpYear	The year the option will expire. Required if type code is OPTN.
CString	m_ExpMonth	The month the option will expire. Required if type code is OPTN.
CString	m_ExpDay	The day the option will expire. Required if type code is OPTN.

List Alerts

```
AlertList CAccountSDK:: GetAlerts()
```

This API lists alerts for the current user. It takes no parameters.

Read Alert

```
CAlertDetailsResponse CAccountSDK:: GetAlertDetails(int nAlertID)
```

This API retrieves alert details.

Type	Name	Description
int	nAlertID	Numeric alert ID

Delete Alert

```
CDeleteAlertResponse CAccountSDK:: DeleteAlert(int nAlertID)
```

This API can be invoked to delete a particular alert.

Type	Name	Description
int	nAlertID	Numeric alert ID

Market Data

The Market library provides methods to access E*TRADE market data.

Get Option Chains

```
COptionChainResponse CMarketSDK:: GetOptionChains(COptionChainRequest  
OptionChainRequestObj)
```

This API returns a list of option chains associated with a specific underlier.

COptionChainRequest

Type	Name	Description
CString	m_ExpirationDay	The day the option will expire
CString	m_ExpirationMonth	The month the option will expire
CString	m_ExpirationYear	The year the option will expire
CString	m_ChainType	The type of option chain. Possible values are: <ul style="list-style-type: none">• CALL (default)• PUT• CALLPUT
CString	m_SkipAdjusted	Value that either shows or does not show adjusted options. Adjusted options are defined as options that have undergone a change and the option contract has been amended to modify this change. Possible values are: TRUE (default), FALSE.
CString	m_Underlier	The market trading symbol for the stock being bought or sold.

Get Option Expiration Dates

```
COptionExpireDateResponse CMarketSDK::
GetOptionExpireDate(COptionExpireDateRequestWrappe
OptionExpireDateRequestObj)
```

This API returns a list of the expiration dates for options that have the specified underlier.

COptionExpireDateRequestWrappe

Type	Name	Description
CString	m_symbol	The symbol of the underlying security or index for the option.

Look Up Product

```
CProductLookupResponse CMarketSDK:: GetProductLookup
(CProductLookupRequest ProductLookupRequestObj)
```

Looks up the exchange and symbol of a security based on the name.

CProductLookupRequest

Type	Name	Description
CString	m_Company	The name of the company.
CString	m_Type	The type of security. Possible values are: <ul style="list-style-type: none">• EQ (equity)• MF (mutual fund)

Get Quote

```
CQuoteResponse CMarketSDK:: GetQuote(CQuoteRegeust QuoteRequestObj)
```

This API returns current market information for equities and options.

CQuoteRegeust

Type	Name	Description
StringList	m_SymbolList	The market trading symbol for the stock being bought or sold. This is a list of symbols which can be any of the following: <ul style="list-style-type: none">• EQ (equity)• OPTN - option The OPTN symbol has the following format (with six components separated by a colon): underlier:year:month:day:optionType:strikePrice

CString	m_DetailFlag	Optional parameter specifying which details to return in the response. The possible values are: <ul style="list-style-type: none"> • ALL (default) • FUNDAMENTAL • INTRADAY • OPTIONS • WEEK_52 The last four return different subsets of the ALL option. Details for all five are listed under the Get Quote API in the REST API Reference.
---------	--------------	---

Orders

The Orders library provides methods to preview, place, modify, and delete market orders.

List Orders

```
COrderListResponse COrderSDK:: GetOrderList(COrderListRequest
OrderListRequestObj)
```

This API returns a list of orders that are currently open or were active during the current day.

COrderListRequest

Type	Name	Description
CString	m_AccountId	Numeric account ID
CString	m_Marker	Parameter that designates additional orders in the account if there are more than the requested number of orders.
CString	m_Count	Parameter that returns the number of list of open orders or orders updated today. If not specified, the default value is 25.

Preview Equity Order

```
CEquityOrderResponse COrderSDK::
GetPreviewEquityOrder(CEquityOrderRequest EquityOrderRequestObj)
```

This API returns a preview of an equity order.

CRequestBase

Type	Name	Description
CString	m_AccountId	Numeric account ID
CString	m_ClientOrderId	Not used when requesting a preview
double	m_LimitPrice	The price at which to buy a share at or below a certain price, or sell the stock when it reaches a specified price. Required if price type is LIMIT or STOP_LIMIT.
long long	m_PreviewId	This property is not used for requesting previews

double	m_StopPrice	The price at which a share is to be bought or sold if specified in a limit order. Required if price type is STOP or STOP_LIMIT.
--------	-------------	---

CBasicOrderRequest

Type	Name	Description
CString	m_AllOrNone	If TRUE, the transactions must be executed all at once, or not at all. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_Quantity	The number of shares to be bought or sold.
CString	m_ReserveOrder	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. If TRUE, must also specify the reserveQuantity. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_ReserveQuantity	The number of shares to be publicly displayed if this is a reserve order. Required if reserve order.

CEquityOrderRequest

Type	Name	Description
CString	m_symbol	The market trading symbol for the share being bought or sold.
CString	m_orderAction	User-specified action that instructs the broker what action to perform. Possible values are: <ul style="list-style-type: none"> • BUY • SELL • BUY_TO_COVER • SELL_SHORT
CString	m_priceType	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE
CString	m_routingDestination	The exchange where a user wants to send orders to be executed. Exchanges that can be selected are: <ul style="list-style-type: none"> • AUTO • ARCA • NSDQ • NYSE
CString	m_marketSession	Session when the equity order will be place. Possible values are: <ul style="list-style-type: none"> • REGULAR • EXTENDED
CString	m_orderTerm	The length of time an equity order is enforced. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only used for limit orders) • FILL_OR_KILL (only used for limit orders)

Place Equity Order

```
CEquityOrderResponse COrderSDK:: GetPlaceEquityorder  
(CEquityOrderRequest EquityOrderRequestObj)
```

This API submits an equity order.

CRequestBase

Type	Name	Description
CString	m_AccountId	Numeric account ID
CString	m_ClientOrderId	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
double	m_LimitPrice	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Required if price type is LIMIT or STOP_LIMIT.
long long	m_PreviewId	If the order was not previewed, this parameter should not be specified. If the order was previewed, this parameter must specify the numeric preview ID from the preview, and other parameters of this request must match the parameters of the preview.
double	m_StopPrice	The price at which a share is to be bought or sold if specified in a limit order. Required if price type is STOP or STOP_LIMIT.

CbasicOrderRequest - inherits from CRequestBase

Type	Name	Description
CString	m_AllOrNone	If TRUE, the transactions must be executed all at once, or not at all. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_Quantity	The number of shares to be bought or sold.
CString	m_ReserveOrder	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. If TRUE, must also specify the reserve quantity. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_ReserveQuantity	The number of shares to be publicly displayed if this is a reserve order. Required if reserve order.

CEquityOrderRequest - inherits from CBasicOrderRequest

Type	Name	Description
CString	m_symbol	The market trading symbol for the security being bought or sold.

CString	m_orderAction	User-specified action that instructs the broker what action to perform. Possible values are: • BUY • SELL • BUY_TO_COVER • SELL_SHORT
CString	m_priceType	The type of pricing. Possible values are: • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE
CString	m_routingDestination	The exchange where a user wants to send orders to be executed. Exchanges that can be selected are: • AUTO • ARCA • NSDQ • NYSE
CString	m_marketSession	Session when the equity order will be placed. Possible values are: REGULAR, EXTENDED.
CString	m_orderTerm	The length of time an equity order is enforced. Possible values are: • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only used for limit orders) • FILL_OR_KILL (only used for limit orders)

Preview Equity Order Change

```
CChangeEquityOrderResponse COrderSDK:: GetPreviewChangeEquityOrder
(CChangeEquityOrderRequest ChangeEquityOrderRequestObj)
```

This API returns a preview of a change to an equity order.

CChangeOrderBase

Type	Name	Description
CString	m_AccountId	Numeric account ID
long long	m_OrderNum	Order number
CString	m_ClientOrderId	Not used when requesting a preview
double	m_LimitPrice	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Required if price type is LIMIT or STOP_LIMIT.
long long	m_PreviewId	This property is not used for requesting previews
double	m_StopPrice	The price at which a share is to be bought or sold if specified in a limit order. Required if price type is STOP or STOP_LIMIT.

CString	m_AllOrNone	If TRUE, the transactions must be executed all at once, or not at all. Possible values are: TRUE, FALSE. Case-insensitive.
CString	m_Quantity	The number of shares to be bought or sold.
CString	m_ReserveOrder	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. If TRUE, must also specify the reserveQuantity. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_ReserveQuantity	The number of shares to be publicly displayed if this is a reserve order. Required if reserve order.

CChangeEquityOrderRequest - inherits from CChangeOrderBase

Type	Name	Description
CString	m_PriceType	The price type. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE
CString	m_OrderTerm	The length of time an equity order is enforced. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only used for limit orders) • FILL_OR_KILL (only used for limit orders)

Place Equity Order Change

CChangeEquityOrderResponse COrderSDK:: GetPlaceChangeEquityOrder

(CChangeEquityOrderRequest ChangeEquityOrderRequestObj)

This API executes a change equity order request.

CChangeOrderBase

Type	Name	Description
CString	m_AccountId	Numeric account ID
long long	m_OrderNum;	Order number
CString	m_ClientOrderId	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
double	m_LimitPrice	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Required if price type is LIMIT or STOP_LIMIT.

long long	m_PreviewId	If the order was not previewed, this parameter should not be specified. If the order was previewed, this parameter must specify the numeric preview ID from the preview, and other parameters of this request must match the parameters of the preview.
double	m_StopPrice	The price at which a share is to be bought or sold if specified in a limit order. Required if price type is STOP or STOP_LIMIT.
CString	m_AllOrNone	If TRUE, the transactions must be executed all at once, or not at all. Possible values are: TRUE, FALSE. Case-insensitive.
CString	m_Quantity	The number of shares to be bought or sold.
CString	m_ReserveOrder	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. If TRUE, must also specify the reserveQuantity. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_ReserveQuantity	The number of shares to be publicly displayed if this is a reserve order. Required if reserve order.

CchangeEquityOrderRequest - CChangeOrderBase

Type	Name	Description
CString	m_PriceType	The price type. Possible values are: <ul style="list-style-type: none"> • MARKET • LIMIT • STOP • STOP_LIMIT • MARKET_ON_CLOSE
CString	m_OrderTerm	The length of time an equity order is enforced. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only used for limit orders) • FILL_OR_KILL (only used for limit orders)

Preview Option Order

```
COptionOrderResponse COrderSDK:: GetPreviewOptionOrder
(COptionOrderRequest OptionOrderRequestObj)
```

This API returns a preview of an option order.

CRequestBase

Type	Name	Description
CString	m_AccountId	Numeric account ID
CString	m_ClientOrderId	Not used when requesting a preview

double	m_LimitPrice	The highest price at which to buy or the lowest price at which to sell if specified in a limit order. Required if price type is LIMIT or STOP_LIMIT.
long long	m_PreviewId	This property is not used for requesting previews
double	m_StopPrice	The price at which to buy or sell if specified in a stop order. Required if priceType is STOP.

CBasicOrderRequest - inherits from CRequestBase

Type	Name	Description
CString	m_AllOrNone	If TRUE, the transactions must be executed all at once, or not at all. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_Quantity	The number of shares to be bought or sold.
CString	m_ReserveOrder	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. If TRUE, must also specify the reserveQuantity. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_ReserveQuantity	The number of shares to be publicly displayed if this is a reserve order. Required if reserve order.

COptionSymbol

Type	Name	Description
CString	m_Symbol	The market trading symbol for the option underlier.
CString	m_CallOrPut	Value that allows the owner of an option the right to either buy or sell an option. Possible values are: CALL, PUT.
double	m_StrikePrice	The price at which an option is exercised
long long	m_ExpirationYear	The year the option will expire
long long	m_ExpirationMonth	The month the option will expire
long long	m_ExpirationDay	The day the option will expire

CoptionOrderRequest - inherits from CBasicOrderRequest

Type	Name	Description
double	m_StopLimitPrice	The designated boundary price for a stop-limit order. Used for STOP_LIMIT orders.
COptionSymbol	m_SymbolInfo	Refer to COptionSymbol
CString	m_OrderAction	User-specified action that instructs the broker what action to perform. Possible values are: <ul style="list-style-type: none"> • BUY_OPEN • SELL_OPEN • BUY_CLOSE • SELL_CLOSE

CString	m_PriceType	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • STOP • LIMIT • STOP_LIMIT
CString	m_RoutingDestination	The exchange where a user wants to send orders to be executed. Exchanges that can be selected are: <ul style="list-style-type: none"> • AUTO • AMEX • BOX • CBOE • ISE • NOM • NYSE • PHX
CString	m_OrderTerm	The length of time an option order is enforced. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only used for limit orders) • FILL_OR_KILL (only used for limit orders)

Place Option Order

```
OptionOrderResponse COrderSDK:: GetPlaceOptionOrder(
OptionOrderRequest OptionOrderRequestObj)
```

This API submits an option order.

CRequestBase

Type	Name	Description
CString	m_AccountId	Numeric account ID
CString	m_ClientOrderId	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
double	m_LimitPrice	The maximum price at which to buy, or the minimum price at which to sell. Required if price type is LIMIT or STOP_LIMIT.
long long	m_PreviewId	If the order was not previewed, this parameter should not be specified. If the order was previewed, this parameter must specify the numeric preview ID from the preview, and other parameters of this request must match the parameters of the preview.
double	m_StopPrice	The price at which to buy or sell if specified in a limit order. Required for STOP orders.

CbasicOrderRequest - inherits from CRequestBase

Type	Name	Description
CString	m_AllOrNone	If TRUE, the transactions must be executed all at once, or not at all. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_Quantity	The number of shares to be bought or sold.
CString	m_ReserveOrder	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. If TRUE, must also specify the reserveQuantity. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_ReserveQuantity	The number of shares to be publicly displayed if this is a reserve order. Required if reserve order.

COptionSymbol

Type	Name	Description
CString	m_Symbol	The market trading symbol for the option underlier
CString	m_CallOrPut	Value that allows the owner of an option the right to either buy or sell an option. Possible values are: CALL, PUT.
double	m_StrikePrice	The strike price for the option
long long	m_ExpirationYear	The year the option will expire
long long	m_ExpirationMonth	The month the option will expire
long long	m_ExpirationDay	The day the option will expire

CoptionOrderRequest - CBasicOrderRequest

Type	Name	Description
double	m_StopLimitPrice	The designated boundary price for a stop-limit order. Used for STOP_LIMIT orders.
COptionSymbol	m_SymbolInfo	Refer to COptionSymbol class definition.
CString	m_OrderAction	User-specified action that instructs the broker what action to perform. Possible values are: <ul style="list-style-type: none">• BUY_OPEN• SELL_OPEN• BUY_CLOSE• SELL_CLOSE
CString	m_PriceType	The type of pricing. Possible values are: <ul style="list-style-type: none">• MARKET• STOP• LIMIT• STOP_LIMIT

CString	m_RoutingDestination	The exchange where a user wants to send orders to be executed. Exchanges that can be selected are: <ul style="list-style-type: none"> • AUTO • AMEX • BOX • CBOE • ISE • NOM • NYSE • PHX
CString	m_OrderTerm	The length of time an option order is enforced. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only used for limit orders) • FILL_OR_KILL (only used for limit orders)

Preview Option Order Change

```
CChangeOptionOrderResponse COrderSDK:: GetPreviewChangeOptionOrder
(CChangeOptionOrderRequest ChangeOptionOrderRequestObj)
```

This API returns a preview of a change to an option order.

CChangeOrderBase

Type	Name	Description
CString	m_AccountId	Numeric account ID
long long	m_OrderNum	Order number
CString	m_ClientOrderId	Not used when requesting a preview
double	m_LimitPrice	The price at which to buy a share at or below a certain price, or sell the stock when it reaches a specified price. Required if price type is LIMIT.
long long	m_PreviewId	This property is not used for requesting previews
double	m_StopPrice	The price at which a contract is to be bought or sold if specified in a limit order. Required if price type is STOP.
CString	m_AllOrNone	If TRUE, the transactions must be executed all at once, or not at all. Possible values are: TRUE, FALSE. Case-insensitive.
CString	m_Quantity	The number of shares to be bought or sold.
CString	m_ReserveOrder	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. If TRUE, must also specify the reserveQuantity. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_ReserveQuantity	The number of shares to be publicly displayed if this is a reserve order. Required if reserve order.

CchangeOptionOrderRequest - inherits from CChangeOrderBase

Type	Name	Description
double	m_StopLimitPrice	The designated boundary price for a stop-limit order. Required if price type is STOP_LIMIT.
CString	m_PriceType	The type of pricing. Possible values are: <ul style="list-style-type: none">• MARKET• STOP• LIMIT• STOP_LIMIT
CString	m_OrderTerm	The length of time an equity order is enforced. Possible values are: <ul style="list-style-type: none">• GOOD_UNTIL_CANCEL• GOOD_FOR_DAY• IMMEDIATE_OR_CANCEL (only used for limit orders)• FILL_OR_KILL (only used for limit orders)

Place Option Order Change

```
CChangeOptionOrderResponse GetPlaceChangeOptionOrder  
(CChangeOptionOrderRequest ChangeOptionOrderRequestObj)
```

This API executes a change to an option order.

CChangeOrderBase

Type	Name	Description
CString	m_AccountId	Numeric account ID
long long	m_OrderNum	Order number
CString	m_ClientOrderId	A reference number generated by the developer. Used to ensure that a duplicate order is not being submitted. It can be any value of 20 alphanumeric characters or less, but must be unique within this account. It does not appear in any API responses.
double	m_LimitPrice	The price at which to buy a share at or below a certain price, or sell the stock when it reaches a specified price. Required if price type is LIMIT.
long long	m_PreviewId	If the order was not previewed, this parameter should not be specified. If the order was previewed, this parameter must specify the numeric preview ID from the preview, and other parameters of this request must match the parameters of the preview.
double	m_StopPrice	The price at which a contract is to be bought or sold if specified in a limit order. Required if price type is STOP.

CString	m_AllOrNone	If TRUE, the transactions must be executed all at once, or not at all. Possible values are: TRUE, FALSE. Case-insensitive.
CString	m_Quantity	The number of shares to be bought or sold.
CString	m_ReserveOrder	If TRUE, publicly displays only a limited number of shares (the reserve quantity), instead of the entire order, to avoid influencing other traders. If TRUE, must also specify the reserveQuantity. Possible values are: TRUE, FALSE. Case-insensitive.
long long	m_ReserveQuantity	The number of shares to be publicly displayed if this is a reserve order. Required if reserve order.

CchangeOptionOrderRequest - inherits from CChangeOrderBase

Type	Name	Description
double	m_StopLimitPrice	The designated boundary price for a stop-limit order. Required if price type is STOP_LIMIT.
CString	m_PriceType	The type of pricing. Possible values are: <ul style="list-style-type: none"> • MARKET • STOP • LIMIT • STOP_LIMIT
CString	m_OrderTerm	The length of time an equity order is enforced. Possible values are: <ul style="list-style-type: none"> • GOOD_UNTIL_CANCEL • GOOD_FOR_DAY • IMMEDIATE_OR_CANCEL (only for limit orders) • FILL_OR_KILL (only for limit orders)

Cancel Order

```
CCancelOrderResponse GetCancelOrder(CcancelOrderRequest
CancelOrderRequestObj)
```

This API submits a cancel request for the indicated order.

CCancelOrderRequest

Type	Name	Description
CString	m_AccountId	Numeric account ID
long long	m_OrderNum	Order number

Error Handling

Common exceptions and error messages for the VC++ SDK.

CExceptionSDK

Type	Name	Description
int	m_nHTTPErrorCode	HTTP error response code
int	m_nErrorCode	SDK error code, as detailed in table of error codes below
CString	m_strErrorMsg	Error message

CError

Type	Name	Description
int	m_nHTTPErrorCode	HTTP error response code.
int	m_nErrorCode	SDK error code, as in table of error codes below
CString	m_strErrorMsg	Error message

SDK Error Codes

Code	Error Description
1001	ERROR_ENVIRONMENT_EMPTY
1002	ERROR_CONSUMER_KEY_EMPTY
1003	ERROR_CONSUMER_SECRET_EMPTY
1004	ERROR_REQUEST_TOKEN_EMPTY
1005	ERROR_REQUEST_TOKEN_SECRET_EMPTY
1006	ERROR_ACCESS_TOKEN_EMPTY
1007	ERROR_ACCESS_TOKEN_SECRET_EMPTY
1008	ERROR_VERIFIER_EMPTY
1009	ERROR_URL_EMPTY
1010	ERROR_POST_PARAM_EMPTY
2000	ERROR_ACCOUNT_ID_EMPTY
2001	ERROR_NO_OF_POSITION_EMPTY
2002	ERROR_ALERT_ID__EMPTY
5000	ERROR_XML_PARSING
3000	ERROR_SYMBOL_EMPTY
3001	ERROR_COMPANY_NAME_EMPTY
3002	ERROR_TYPE_EMPTY
3003	ERROR_SYMBOL_KEY_EMPTY
5000	ERROR_MISSING_REQUIRED_FIELD

Code Snippets: Accounts

This section contains useful code for working with the Accounts API module.

Initialize Accounts Client

This code should be run once before using the other Accounts snippets, after authorizing the application.

PHP Code Snippet

```
$consumer = new etOAuthConsumer('Your consumer key','Your secret');
$consumer->oauth_token = 'Your access token';
$consumer->oauth_token_secret = 'Your token secret';
$ac_obj = new etAccounts($consumer);
```

Java Code Snippet

```
ClientRequest request = new ClientRequest();
request.setEnv(Environment.LIVE); // set to LIVE or SANDBOX
request.setConsumerKey("Your consumer key");
request.setConsumerSecret("Your consumer secret");
request.setToken("Your access token");
request.setTokenSecret("Your token secret");
```

List Accounts

This code retrieves a list of the user's E*TRADE accounts.

PHP Code Snippet

```
$ac_list = $ac_obj->GetAccountList();
```

Java Code Snippet

```
AccountsClient client = new AccountsClient(request);
AccountListResponse response = client.getAccountList();
```

Get Account Balance

This code retrieves the current account balance and related details for a specified account.

PHP Code Snippet

```
$ac_balance = $ac_obj->GetAccountBalance('Your account ID');
```

Java Code Snippet

```
AccountBalanceResponse balance = client.getAccountBalance("Your account ID");
```

Get Account Positions

This code retrieves a list of the positions held in the specified account.

PHP Code Snippet

```
//Construct params object
$request_params = new AccountPositionsRequest();
$request_params->__set(count, your_count);
$request_params->__set(expDay, your_day);
//Call GetAccountPositions
$sac_positions = $sac_obj->GetAccountPositions($sac_id, $request_params);
```

Java Code Snippet

```
AccountPositionsResponse aprs = null;
AccountPositionsRequest apr = new AccountPositionsRequest();
apr.setCount(10); // count is set to 10
apr.setMarker("Your marker value"); // insert marker
apr.setSymbol("Your symbol value"); // insert desired symbol
apr.setTypeCode("Your type code"); // set type code to EQ, OPTN, MF, or BOND
aprs = client.getAccountPositions(acct, apr);
```

List Alerts

This code retrieves all alerts for the user.

PHP Code Snippet

```
$sac_alerts = $sac_obj->GetAlerts();
```

Java Code Snippet

```
GetAlertsResponse alerts = client.getAlerts();
```

Read Alert

This code retrieves detail on a specified alert.

PHP Code Snippet

```
$sac_alert_dtl = $sac_obj->GetAlertDetails('Your alert ID');
```

Java Code Snippet

```
GetAlertDetailsResponse details = client.getAlertDetail("Your alert ID");
```

Delete Alert

This code deletes a specified alert.

PHP Code Snippet

```
$ac_alert_del = $ac_obj->DeleteAlert('Your alert ID');
```

Java Code Snippet

```
DeleteAlertResponse deleteAlert = client.deleteAlert("Your alert ID");
```

Code Snippets: Market

This section contains useful code snippets for working with the Market API module.

Initialize Market Client

This code should be run once before using the other Market snippets, after authorizing the application.

PHP Code Snippet

```
$consumer = new etOAuthConsumer('Your consumer key','Your consumer secret');
$consumer->oauth_token = 'Your access token';
$consumer->oauth_token_secret = 'Your token secret';
$mc_obj = new MarketClient($consumer);
```

Java Code Snippet

```
ClientRequest request = new ClientRequest();
request.setEnv(Environment.LIVE);
request.setConsumerKey("Your consumer key");
request.setConsumerSecret("Your consumer secret");
request.setToken("Your access token");
request.setTokenSecret("Your token secret");
```

Get Quote

These code snippets return current market information for equities and options.

PHP Code Snippet

```
$request_params = new getQuoteParams();
$request_params->__set('symbolList', array('GOOG')); // symbolList = GOOG for example
$request_params->__set('detailFlag', 'WEEK_52'); // detailFlag = WEEK_52 for example
$out = $mc_obj->getQuote($request_params);
```

Java Code Snippet

```
ArrayList<String> list = new ArrayList<String>();
MarketClient client = new MarketClient(request);
list.add("CSCO");
list.add("AAPL");
QuoteResponse response = client.getQuote(list, new Boolean(afterHourFlag),
DetailFlag.ALL);
```

Look Up Product

Looks up the exchange and symbol of a security based on the name.

PHP Code Snippet

```
$request_params = new productLookupParams();
$request_params->__set('company', 'cisco'); // company = "cisco" for example
$request_params->__set('type', 'eq'); // type = equity for example
$out= $mc_obj->productLookup($request_params);
```

Java Code Snippet

```
ProductLookupRequest req = new ProductLookupRequest();
req.setCompany("Your company name");
req.setType(SecurityType.EQ.value()); // OR SecurityType.INDX.value();
ProductLookupResponse response = client.productLookup(req);
```

Get Option Expire Dates

This code retrieves a list of the expiration dates for options that have the specified underlier.

PHP Code Snippet

```
$request_params = new getExpiryDateParams();
$request_params->__set('underlier', 'GOOG'); // underlier = GOOG for example
$out= $mc_obj->getExpiryDates($request_params);
```

Java Code Snippet

```
OptionExpireDateGetRequest req = new OptionExpireDateGetRequest();
req.setUnderlier("GOOG"); // underlier = GOOG for example
OptionExpireDateGetResponse response = client.getExpiryDates(req);
```

Get Option Chains

This code retrieves a list of option chains associated with a specific underlier.

PHP Code Snippet

```
$request_params = new getOptionChainsParams();
$request_params->__set('expirationMonth', '1'); // example values
$request_params->__set('expirationYear', '2012');
$request_params->__set('chainType', 'PUT');
$request_params->__set('skipAdjusted', 'TRUE');
$request_params->__set('underlier', 'GOOG');
$out= $mc_obj->getOptionChain($request_params);
```

Java Code Snippet

```
OptionChainRequest req = new OptionChainRequest();
req.setExpirationMonth("1"); // example values
req.setExpirationYear("2012");
req.setChainType("CALL"); // example values
req.setSkipAdjusted("FALSE");
req.setUnderlier("GOOG");
OptionChainResponse response = client.getOptionChain(req);
```


Code Snippets: Order

This section contains useful code snippets for working with the Order API module.

Initialize Order Client

This code should be run once before using the other Order snippets, after authorizing the application.

PHP Code Snippet

```
$consumer = new etOAuthConsumer('Your consumer key','Your consumer secret');  
$consumer->oauth_token = 'Your access token';  
$consumer->oauth_token_secret = 'Your token secret';  
$mc_obj = new OrderClient($consumer);
```

Java Code Snippet

```
ClientRequest request = new ClientRequest();  
request.setEnv(Environment.LIVE);  
request.setConsumerKey("Your consumer key");  
request.setConsumerSecret("Your consumer secret");  
request.setToken("Your access token");  
request.setTokenSecret("Your token secret");
```

List Orders

This code returns a list of orders that are currently open or have had activity during the current day.

PHP Code Snippet

```
$out = $oc_obj->getOrderList(83405188);
```

Java Code Snippet

```
OrderClient client = new OrderClient(request);  
OrderListRequest olr = new OrderListRequest();  
olr.setAccountId("83405188"); // sample values  
olr.setCount(25);  
olr.setMarker("1270023312|296");  
GetOrderListResponse response = client.getOrderList(olr);
```

Preview Equity Order

This code creates a preview of an equity order, useful for validating inputs and verifying the cost before placing the order.

PHP Code Snippet

```
$request_params = new EquityOrderRequest();
$request_params->__set('accountId',83405188); // sample values
$request_params->__set('clientOrderId','asdf1234');
$request_params->__set('limitPrice',300);
$request_params->__set('previewId','');
$request_params->__set('stopPrice',300);
$request_params->__set('allOrNone','');
$request_params->__set('quantity',4);
$request_params->__set('reserveOrder','');
$request_params->__set('reserveQuantity',0);
$request_params->__set('stopLimitPrice','');
$request_params->__set('symbol','AAPL');
$request_params->__set('orderAction','BUY');
$request_params->__set('priceType','LIMIT');
$request_params->__set('routingDestination','');
$request_params->__set('marketSession','REGULAR');
$request_params->__set('orderTerm','GOOD_FOR_DAY');

$request_xml_object = new PreviewEquityOrder($request_params);
$out = $oc_obj->previewEquityOrder($request_xml_object);
```

Java Code Snippet

```
PreviewEquityOrder orderRequest = new PreviewEquityOrder();
EquityOrderRequest eor = new EquityOrderRequest();
eor.setAccountId("83405188"); // sample values
eor.setSymbol("AAPL");
eor.setAllOrNone("FALSE");
eor.setClientOrderId("asdf1234");
eor.setOrderTerm(EquityOrderTerm.GOOD_FOR_DAY);
eor.setOrderAction(EquityOrderAction.BUY);
eor.setMarketSession(MarketSession.REGULAR);
eor.setPriceType(EquityPriceType.MARKET);
eor.setQuantity(new BigInteger("100"));
eor.setRoutingDestination(EquityOrderRoutingDestination.AUTO.value());
eor.setReserveOrder("TRUE");
orderRequest.setEquityOrderRequest(eor);
PreviewEquityOrderResponse response = client.previewEquityOrder(orderRequest);
```

Place Equity Order

This code places an equity order.

PHP Code Snippet

```
$request_params = new EquityOrderRequest();
$request_params->__set('accountId',83405188); // sample values
$request_params->__set('clientId','asdf1234');
$request_params->__set('limitPrice',300);
$request_params->__set('previewId','');
$request_params->__set('stopPrice',300);
$request_params->__set('allOrNone','');
$request_params->__set('quantity',4);
$request_params->__set('reserveOrder','');
$request_params->__set('reserveQuantity',0);
$request_params->__set('stopLimitPrice','');
$request_params->__set('symbol','AAPL');
$request_params->__set('orderAction','BUY');
$request_params->__set('priceType','LIMIT');
$request_params->__set('routingDestination','');
$request_params->__set('marketSession','REGULAR');
$request_params->__set('orderTerm','GOOD_FOR_DAY');

$request_xml_object = new PlaceEquityOrder($request_params);
$out = $oc_obj->placeEquityOrder($request_xml_object);
```

Java Code Snippet

```
PlaceEquityOrder orderRequest = new PlaceEquityOrder();
EquityOrderRequest eor = new EquityOrderRequest();
eor.setAccountId("83405188"); // sample values
eor.setSymbol("AAPL");
eor.setAllOrNone("FALSE");
eor.setClientId("asdf1234");
eor.setOrderTerm(EquityOrderTerm.GOOD_FOR_DAY);
eor.setOrderAction(EquityOrderAction.BUY);
eor.setMarketSession(MarketSession.REGULAR);
eor.setPriceType(EquityPriceType.LIMIT);
eor.setLimitPrice(new BigDecimal("1.5"));
eor.setQuantity(new BigInteger("3"));
eor.setRoutingDestination(EquityOrderRoutingDestination.AUTO.value());
eor.setReserveOrder("FALSE");
orderRequest.setEquityOrderRequest(eor);
PlaceEquityOrderResponse response = client.placeEquityOrder(orderRequest);
```

Preview Equity Order Change

This code creates a preview of a change to an equity order, useful for validating inputs and verifying costs before actually requesting the change.

PHP Code Snippet

```
$request_params = new changeEquityOrderRequest();
$request_params->__set('priceType',''); // sample values
```

```

$request_params->__set('orderTerm','GOOD_UNTIL_CANCEL');
$request_params->__set('accountId',83405188);
$request_params->__set('orderNum',162);
$request_params->__set('clientOrderId','asdf1234');
$request_params->__set('limitPrice','');
$request_params->__set('previewId','');
$request_params->__set('stopPrice','');
$request_params->__set('allOrNone','');
$request_params->__set('quantity','');
$request_params->__set('reserveOrder','');
$request_params->__set('reserveQuantity','');

$request_xml_object = new PreviewChangeEquityOrder($request_params);
$out = $oc_obj->previewChangeEquityOrder($request_xml_object);

```

Java Code Snippet

```

PreviewChangeEquityOrder orderRequest = new PreviewChangeEquityOrder();
ChangeEquityOrderRequest eor = new ChangeEquityOrderRequest();
eor.setAccountId("83405188"); // sample values
eor.setOrderNum(new BigInteger("162"));
eor.setAllOrNone("FALSE");
eor.setClientOrderId("asdf1234");
eor.setOrderTerm(EquityOrderTerm.GOOD_FOR_DAY.value());
eor.setPriceType(EquityPriceType.MARKET.value());
eor.setQuantity("4");
eor.setReserveOrder("FALSE");
orderRequest.setChangeEquityOrderRequest(eor);
PreviewChangeEquityOrderResponse response =
client.previewChangeEquityOrder(orderRequest);

```

Place Equity Order Change

This code places the change request to modify an existing equity order.

PHP Code Snippet

```

$request_params = new changeEquityOrderRequest();
$request_params->__set('priceType',''); // sample values
$request_params->__set('orderTerm','GOOD_UNTIL_CANCEL');
$request_params->__set('accountId',83405188);
$request_params->__set('orderNum',162);
$request_params->__set('clientOrderId','asdf1234');
$request_params->__set('limitPrice','');
$request_params->__set('previewId','');
$request_params->__set('stopPrice','');
$request_params->__set('allOrNone','');
$request_params->__set('quantity','');
$request_params->__set('reserveOrder','');
$request_params->__set('reserveQuantity','');

```

```
$request_xml_object = new PlaceChangeEquityOrder($request_params);
$out = $oc_obj->placeChangeEquityOrder($request_xml_object);
```

Java Code Snippet

```
PlaceChangeEquityOrder orderRequest = new PlaceChangeEquityOrder();
ChangeEquityOrderRequest eor = new ChangeEquityOrderRequest();
eor.setAccountId("83405188"); // sample values
eor.setOrderNum(new BigInteger("162"));
eor.setAllOrNone("FALSE");
eor.setClientOrderId("asdf1234");
eor.setOrderTerm(EquityOrderTerm.GOOD_FOR_DAY.value());
eor.setPriceType(EquityPriceType.LIMIT.value());
eor.setLimitPrice(new BigDecimal("1.5"));
eor.setQuantity("4");
eor.setReserveOrder("FALSE");
//eor.setPreviewId(new BigInteger("449548422022"));
orderRequest.setChangeEquityOrderRequest(eor);
PlaceChangeEquityOrderResponse response = client.placeChangeEquityOrder(orderRequest);
```

Preview Option Order

This code creates a preview of an option order, useful for validating inputs and verifying the cost before placing the order.

PHP Code Snippet

```
//Build option_symbol_obj
$option_symbol_obj = new OptionSymbol();
$option_symbol_obj->__set('symbol','AAPL'); // sample values
$option_symbol_obj->__set('callOrPut','CALL');
$option_symbol_obj->__set('strikePrice',115);
$option_symbol_obj->__set('expirationYear','2011');
$option_symbol_obj->__set('expirationMonth','11');
$option_symbol_obj->__set('expirationDay','17');

//Build request_params
$request_params = new OptionOrderRequest();
$request_params->__set('accountId',83405188);
$request_params->__set('clientOrderId',"asdf1234");
$request_params->__set('limitPrice',3);
$request_params->__set('previewId','');
$request_params->__set('stopPrice','');
$request_params->__set('allOrNone','');
$request_params->__set('quantity',4);
$request_params->__set('reserveOrder','');
$request_params->__set('reserveQuantity','');
$request_params->__set('symbolInfo',$option_symbol_obj);
$request_params->__set('stopLimitPrice','');
$request_params->__set('orderAction','BUY_OPEN');
$request_params->__set('priceType','LIMIT');
```

```

$request_params->__set('marketSession','REGULAR');
$request_params->__set('orderTerm','GOOD_FOR_DAY');
$request_params->__set('routingDestination','');

$request_xml_object = new PreviewOptionOrder($request_params);
$out = $oc_obj->previewOptionOrder($request_xml_object);

```

Java Code Snippet

```

PreviewOptionOrder orderRequest = new PreviewOptionOrder();
OptionOrderRequest oor = new OptionOrderRequest();
oor.setAccountId("83405188");
oor.setAllOrNone("FALSE");
oor.setClientOrderId("asdf1234");
oor.setOrderAction(OptionOrderAction.BUY_OPEN);
oor.setOrderTerm(OptionOrderTerm.GOOD_FOR_DAY);
oor.setPriceType(OptionPriceType.MARKET);
oor.setQuantity(new BigInteger("4"));
oor.setReserveOrder("FALSE");
oor.setRoutingDestination(OptionOrderRoutingDestination.AUTO.value());
OptionSymbol os = new OptionSymbol();
os.setCallOrPut(CallOrPut.CALL);
os.setExpirationDay(BigInteger.valueOf(10));
os.setExpirationMonth(BigInteger.valueOf(12));
os.setExpirationYear(BigInteger.valueOf(2010));
os.setStrikePrice(new BigDecimal("1.5"));
os.setSymbol("AAPL");
oor.setSymbolInfo(os);
orderRequest.setOptionOrderRequest(oor);
PreviewOptionOrderResponse response = client.previewOptionOrder(orderRequest);

```

Place Option Order

This code places an option order.

PHP Code Snippet

```

//Build option_symbol_obj
$option_symbol_obj = new OptionSymbol();
$option_symbol_obj->__set('symbol','AAPL'); // sample values
$option_symbol_obj->__set('callOrPut','CALL');
$option_symbol_obj->__set('strikePrice',115);
$option_symbol_obj->__set('expirationYear','2011');
$option_symbol_obj->__set('expirationMonth','11');
$option_symbol_obj->__set('expirationDay','17');

//Build request_params
$request_params = new OptionOrderRequest();

$request_params->__set('accountId',83405188);
$request_params->__set('clientOrderId',"asdf1234");

```

```

$request_params->__set('limitPrice',3);
$request_params->__set('previewId','');
$request_params->__set('stopPrice','');
$request_params->__set('allOrNone','');
$request_params->__set('quantity',4);
$request_params->__set('reserveOrder','');
$request_params->__set('reserveQuantity','');
$request_params->__set('symbolInfo',$option_symbol_obj);
$request_params->__set('stopLimitPrice','');
$request_params->__set('orderAction','BUY_OPEN');
$request_params->__set('priceType','LIMIT');
$request_params->__set('marketSession','REGULAR');
$request_params->__set('orderTerm','GOOD_FOR_DAY');
$request_params->__set('routingDestination','');

$request_xml_object = new PlaceOptionOrder($request_params);
$out = $oc_obj->placeOptionOrder($request_xml_object);

```

Java Code Snippet

```

PlaceOptionOrder orderRequest = new PlaceOptionOrder();
OptionOrderRequest oor = new OptionOrderRequest();
oor.setAccountId("83405188"); // sample values
oor.setAllOrNone("FALSE");
oor.setClientOrderId("asdf1234");
oor.setOrderAction(OptionOrderAction.BUY_OPEN);
oor.setOrderTerm(OptionOrderTerm.GOOD_FOR_DAY);
oor.setPriceType(OptionPriceType.MARKET);
oor.setQuantity(BigInteger.valueOf(3));
oor.setReserveOrder("FALSE");
oor.setRoutingDestination(OptionOrderRoutingDestination.AUTO.value());
OptionSymbol os = new OptionSymbol();
os.setCallOrPut(CallOrPut.CALL);
os.setExpirationDay(BigInteger.valueOf(20));
os.setExpirationMonth(BigInteger.valueOf(12));
os.setExpirationYear(BigInteger.valueOf(2010));
os.setStrikePrice(new BigDecimal("470"));
os.setSymbol("AAPL");
oor.setSymbolInfo(os);
orderRequest.setOptionOrderRequest(oor);
PlaceOptionOrderResponse response = client.placeOptionOrder(orderRequest);

```

Preview Option Order Change

This code creates a preview of a change to an option order, useful for validating inputs and verifying costs before actually requesting the change.

PHP Code Snippet

```

$request_params = new changeOptionOrderRequest();
$request_params->__set('stopLimitPrice',''); // sample values

```

```

$request_params->__set('priceType','');
$request_params->__set('orderTerm','GOOD_UNTIL_CANCEL');
$request_params->__set('accountId',83405188);
$request_params->__set('orderNum',162);
$request_params->__set('clientOrderId','asdf1234');
$request_params->__set('limitPrice','');
$request_params->__set('previewId','');
$request_params->__set('stopPrice','');
$request_params->__set('allOrNone','');
$request_params->__set('quantity','');
$request_params->__set('reserveOrder','');
$request_params->__set('reserveQuantity','');

$request_xml_object = new PreviewChangeOptionOrder($request_params);
$out = $oc_obj->previewChangeOptionOrder($request_xml_object);

```

Java Code Snippet

```

PreviewChangeOptionOrder orderRequest = new PreviewChangeOptionOrder();
ChangeOptionOrderRequest oor = new ChangeOptionOrderRequest();
oor.setAccountId("83405188");
oor.setOrderNum(new BigInteger("162"));
oor.setAllOrNone("FALSE");
oor.setClientOrderId("asdf1234");
oor.setOrderTerm(OptionOrderTerm.GOOD_FOR_DAY.value());
oor.setPriceType(OptionPriceType.MARKET.value());
oor.setQuantity("5");
oor.setReserveOrder("FALSE");
orderRequest.setChangeOptionOrderRequest(oor);
PreviewChangeOptionOrderResponse response =
client.previewChangeOptionOrder(orderRequest);
ChangeOptionOrderResponse opt = response.getOptionOrderResponse();

```

Place Option Order Change

This code places the change request to modify an existing option order.

PHP Code Snippet

```

$request_params = new changeEquityOrderRequest();
$request_params->__set('priceType','');
$request_params->__set('orderTerm','GOOD_UNTIL_CANCEL');
$request_params->__set('accountId',83405188);
$request_params->__set('orderNum',162);
$request_params->__set('clientOrderId','asdf1234');
$request_params->__set('limitPrice','');
$request_params->__set('previewId','');
$request_params->__set('stopPrice','');
$request_params->__set('allOrNone','');
$request_params->__set('quantity','');
$request_params->__set('reserveOrder','');

```



```

$request_params->__set('reserveQuantity','');

$request_xml_object = new PlaceChangeEquityOrder($request_params);
$out = $oc_obj->placeChangeEquityOrder($request_xml_object);

```

Java Code Snippet

```

PlaceChangeOptionOrder orderRequest = new PlaceChangeOptionOrder();
ChangeOptionOrderRequest oor = new ChangeOptionOrderRequest();
oor.setAccountId("83405188");
oor.setOrderNum(new BigInteger("162"));
oor.setAllOrNone("FALSE");
oor.setClientOrderId("asdf1234");
oor.setOrderTerm(OptionOrderTerm.GOOD_FOR_DAY.value());
oor.setPriceType(OptionPriceType.MARKET.value());
oor.setQuantity("4");
oor.setReserveOrder("FALSE");
orderRequest.setChangeOptionOrderRequest(oor);
PlaceChangeOptionOrderResponse response = client.placeChangeOptionOrder(orderRequest);

```

Cancel Order

This code requests an order cancellation.

PHP Code Snippet

```

$request_params = new CancelOrderRequest();
$request_params->__set('accountId',83405188);
$request_params->__set('orderNum',162);

$request_xml_object = new CancelOrder($request_params);
$out = $oc_obj->cancelOrder($request_xml_object);

```

Java Code Snippet

```

CancelOrderRequest corequest = new CancelOrderRequest();
corequest.setAccountId("83405188");
corequest.setOrderNum(new BigInteger("162"));
CancelOrder corder = new CancelOrder();
corder.setCancelOrderRequest(corequest);
OrderClient oclient = new OrderClient(request);
CancelOrderResponse response = oclient.cancelOrder(corder);
CancelResponse cresponse = response.getCancelResponse();

```