EveKit Documentation

Release 4.2.1

Orbital Enterprises

Oct 02, 2018

User Guide

EveKit is a web-based toolkit for building applications based on the EVE Online third party APIs. The primary function of EveKit is to periodically retrieve and store your character and corporation data using the EVE Swagger Interface. EveKit attempts to retrieve your data as frequently as the ESI timers allow. Updates to your data are stored in a time series, allowing for both current and historical views. You can access your data directly via our REST API; or, you can use a third party tool which supports EveKit.

CHAPTER 1

Features

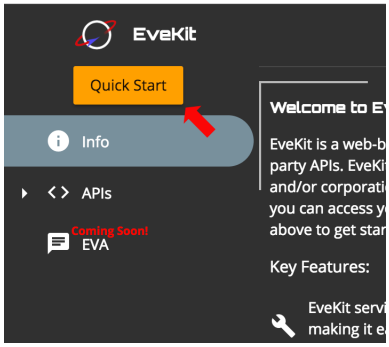
EveKit was designed to provide the following key features:

- **Web Ready**: EveKit is designed for easy access via common web protocols. Every API is annotated with Swagger markup, making it very easy to work directly with our REST services (resturning JSON). You can also generate a client in the language of your choice using free tools like the Swagger Editor.
- **History**: What differentiates EveKit from other EVE Online tools is the retention of a complete history of all your account data. Each aspect of your account data is organized into a time series. This means that we version your data so that each time EveKit performs an update (attempted as frequently as the EVE API cache timers allow), we store any changes in a new version. An API argument let's you decide whether you want to look at the latest version of your data, or some version recorded in the past.
- **Meta-Data**: Account data we store on your behalf can be tagged with persistent meta data. 'Persistent' means this data is preserved across all versions of your account data. Meta data makes it possible to store custom data you might need for certain applications. In particular, meta data simplifies the creation of EveKit extensions (documented elsewhere).
- **Bulk Snapshots**: The latest version of all of your account data is always available to you in bulk form (as an archive of CSV files). We recognize that not everyone will want to use our APIs to access their data, so we generate daily snapshots which you can access from the account summary.
- **Related APIs**: EveKit also provides APIs for important public data such as ESI reference data (for example, alliance membership), the Static Data Export (SDE), and market data. Some of these APIs are exposed via REST, while others are exposed as bulk file downloads.

CHAPTER 2

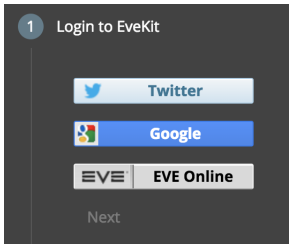
Getting Started

Getting started with EveKit is as easy as clicking on the "QuickStart" button on the main page:



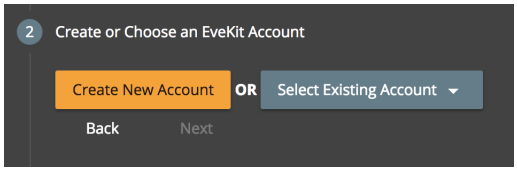
The quickstart process will walk you through the following steps:

1. **Login:** The first step is to authenticate to EveKit using one of the supported authentication services:



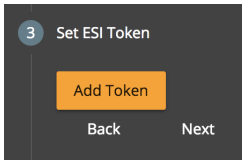
   All EVE accounts you ask EveKit to store will be associated with the user you authenticate as in this step. Note that your login credentials *are not* used to authenticate with the ESI servers for retrieving account data. The credentials needed for ESI authentication are provided in step 3.

2. **Create Account:** The next step is to create your first EveKit account. Select the 'Create Account' button, type in a name for your new account, then specify whether this will be a 'CHARACTER' account (an account which records EVE character information), or a 'CORPORATION' account (an account which records EVE corporation information):



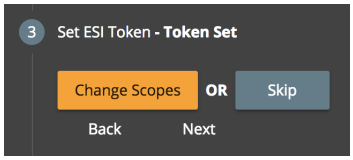
Account names may be composed of letters, digits or underscores, and can be modified later as needed. The account type (character or corporation) **can not** be changed once specified. If you make a mistake, you can always delete the account and create a new one. If this is not your first account, you may notice that it is also possible to select an existing account. You can ignore that option for now, we'll cover it later in this section.

3. **Set ESI Token:** Your new account needs an ESI token which allows EveKit to access your data. Click on the 'Add Token' button to start the ESI token dialog:



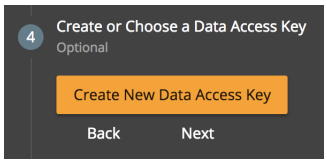
This dialog allows you to select all scopes which should be made available to EveKit. Next to each scope name is a description of the data which can be accessed using that scope. You can select as many scopes as you like. Note that you can revoke your tokens from EveKit at any time using the ESI Token Management Site. Once you are satisfied with your scope selection, click the 'Add Credential' button to start the authentication flow with EVE Online. This flow allows you to select which EVE Online character will be used for access. You can also verify the scopes you are making available to EveKit.

If you complete the authentication flow successfully, you will return to the EveKit site where step 3 will now look as follows:



If you're satisfied with your scope selection, click the 'Skip' button. Otherwise, you can select 'Change Scopes' to change your scope selection.

4. **Create Access Key (optional):** At this point, EveKit will start collecting data for your new account. If you'd like to access your data, however, either through the EveKit main page or via a third party library, then you'll need to create a data access key:



Click the 'Create New Data Access Key' button to get started. Just like ESI scopes, EveKit allows you to permission access keys so that only certain account data is accessible to the user of the key. You'll need to specify a name for the key (letters, digits, or underscores), a date when the key will expire (or 'Never'), a date which limits the how far back someone can look at account data (or 'Unlimited' - more on this below), and, finally, the set of permissions to assign to the key. Next to each permission is a list of the data allowed by the given permission. When you're satisfied with your selections, click 'Save Key'.

Congratulations! You've just created your first EveKit account! Clicking the 'Finish' key will take you back to the EveKit landing page. If you'd like to create more accounts, you can run the quickstart process again. There is also an 'Add New...' option under the 'Accounts' menu section.

2.1 Frequently Asked Questions about Account Creation

How many accounts can I create? At present, there is no limit on the number of accounts which can be associated with a login. We may impose a limit in the future if this is abused.

How many access keys can I create? At present, there is no limit on the number of access keys which can be attached to an account. We may impose a limit in the future if this is abused.

How long does it take to start collecting data for my account? At current loads, a new account will start collecting data about ten minutes after the ESI key is set.

I messed up an account, how do I fix it? The easiest way to start over is to delete the unwanted account and create a new one. You can delete an existing account by selecting the account name in the 'Account' menu. Click the

trash can icon in the summary view and confirm you want to delete the account. To prevent accidents, EveKit will not actually remove the account for 24 hours. You can restore the account at any time before then.

I want to use the same account name for an account I just deleted. How do I do that? You'll need to rename the account you just deleted. You can do that by first restoring the deleted account: go to the summary view for the account and click the red trash can with the up arrow. This will restore the account. Now click the rename button (pencil icon) and change the name of the account. The previous account name is now usable for new accounts. You can now delete the old account again.

CHAPTER 3

Interface Overview

The following figure highlights key parts of the EveKit interface (after login):

The left side of the interface contains the navigation menu. This menu wil change over time as new features are added to EveKit. The 'Accounts' section will list any accounts you add to EveKit. The 'APIs' section will allow you to browse your account data, as well as other data EveKit provides (such as the Static Data Export, reference data, and market data). The API browser is just a wrapper around the Swagger UI. That is, browsing the APIs consists of making REST calls via the Swagger UI. The center portion of the interface always displays the currently selected menu entry.

The top right portion of the interface allows access to configuration settings, any notifications provided by the EveKit backend, as well as access to information about the currently logged in user. This interface is described in more detail

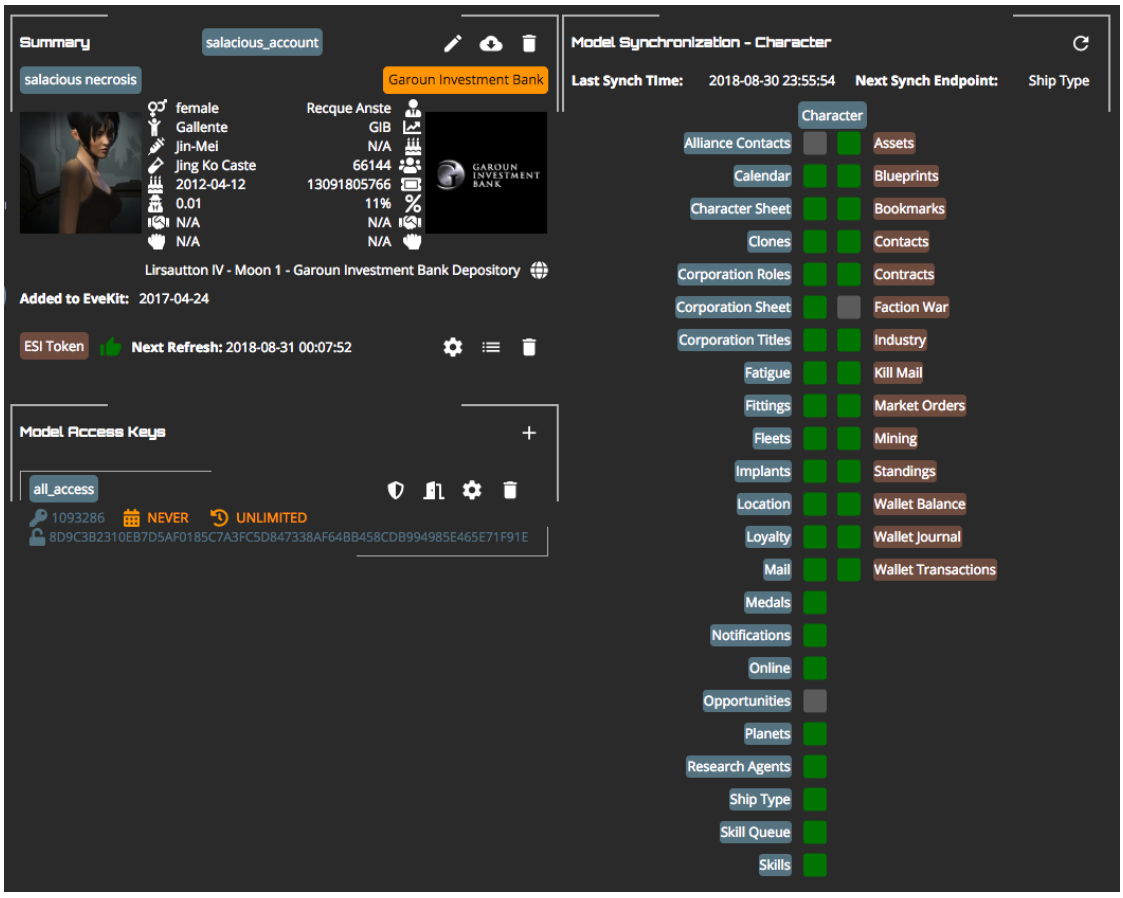
in the Settings, Notifications, and User Information section.

Finally, links on the lower left of the interface allow easy access to the EveKit YouTube channel, and our blog.

CHAPTER 4

Account Screen

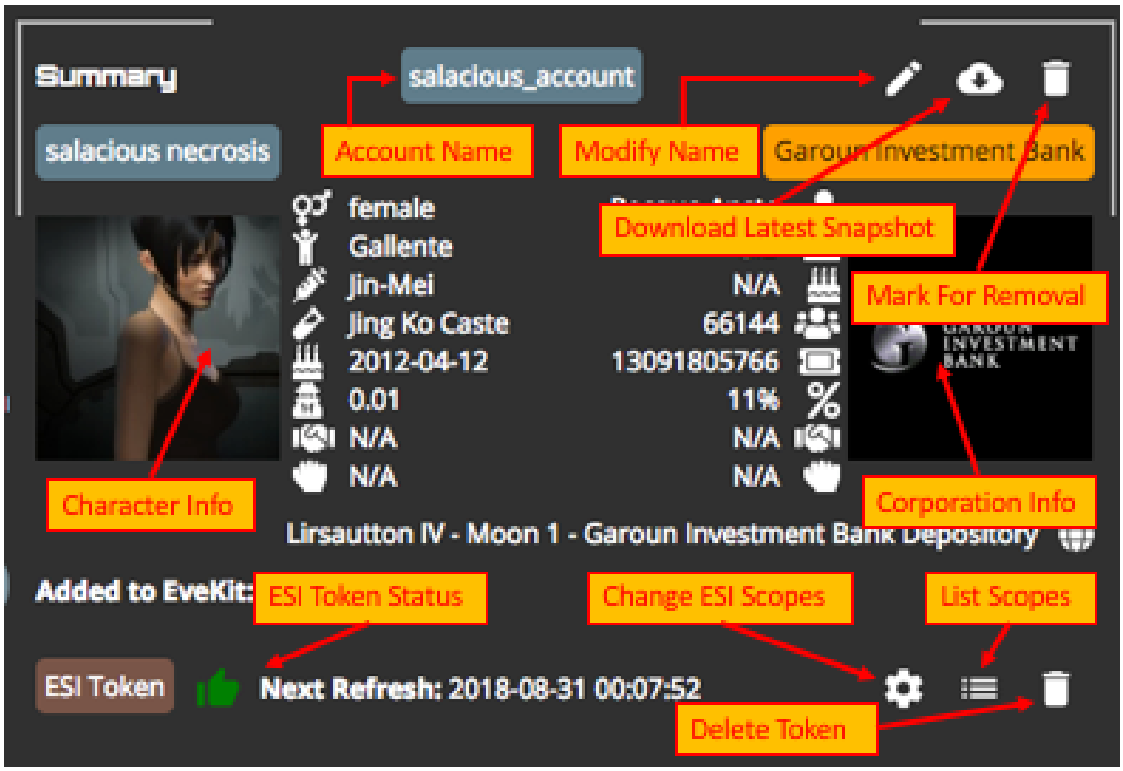
An account screen is generated for each account you have registered with EveKit. A typical screen appears as follows:



We'll explore each section in more detail below.

4.1 Summary

The account summary displays basic information about the account and provides controls for managing the ESI token used to access EVE Online data:



Summary information is provided regardless of whether this account will record character or corporation data. The account name is listed at the top followed by controls for modifying the account name (pencil icon), downloading the latest account snapshot (cloud icon), and marking the account for removal (garbage can icon). As noted elsewhere, accounts are not deleted until 24 hours after they are marked for removal. This gives you ample opportunity to change your mind.

Public information about the character and corporation associated with the account is provided in the middle of the summary.

The bottom of the summary displays information about the ESI token used to access data from EVE Online's servers. If the token is still valid, then a green "thumbs up" icon will be shown along with the date and time when the EveKit backend will next need to refresh the token. If the token is not valid, then a red "thumbs down" icon will appear along with a button for re-authorizing the token (not shown). While the token is invalid, only public character and corporation data will be recorded. You'll need to re-authorize an invalid token if you wish to continue recording non-public data.

On the right side of the token display, there are controls which allow you to change token scopes, list the current scopes, or remove the token completely. If you remove the ESI token, then data will no longer be recorded for this account. However, you can continue to access any saved data (using the model API) until you delete the account.

4.2 Model Access Keys

The model access keys section displays all EveKit API keys which can be used to access the data recorded for this account:

The plus icon on the top right is used to add new keys.

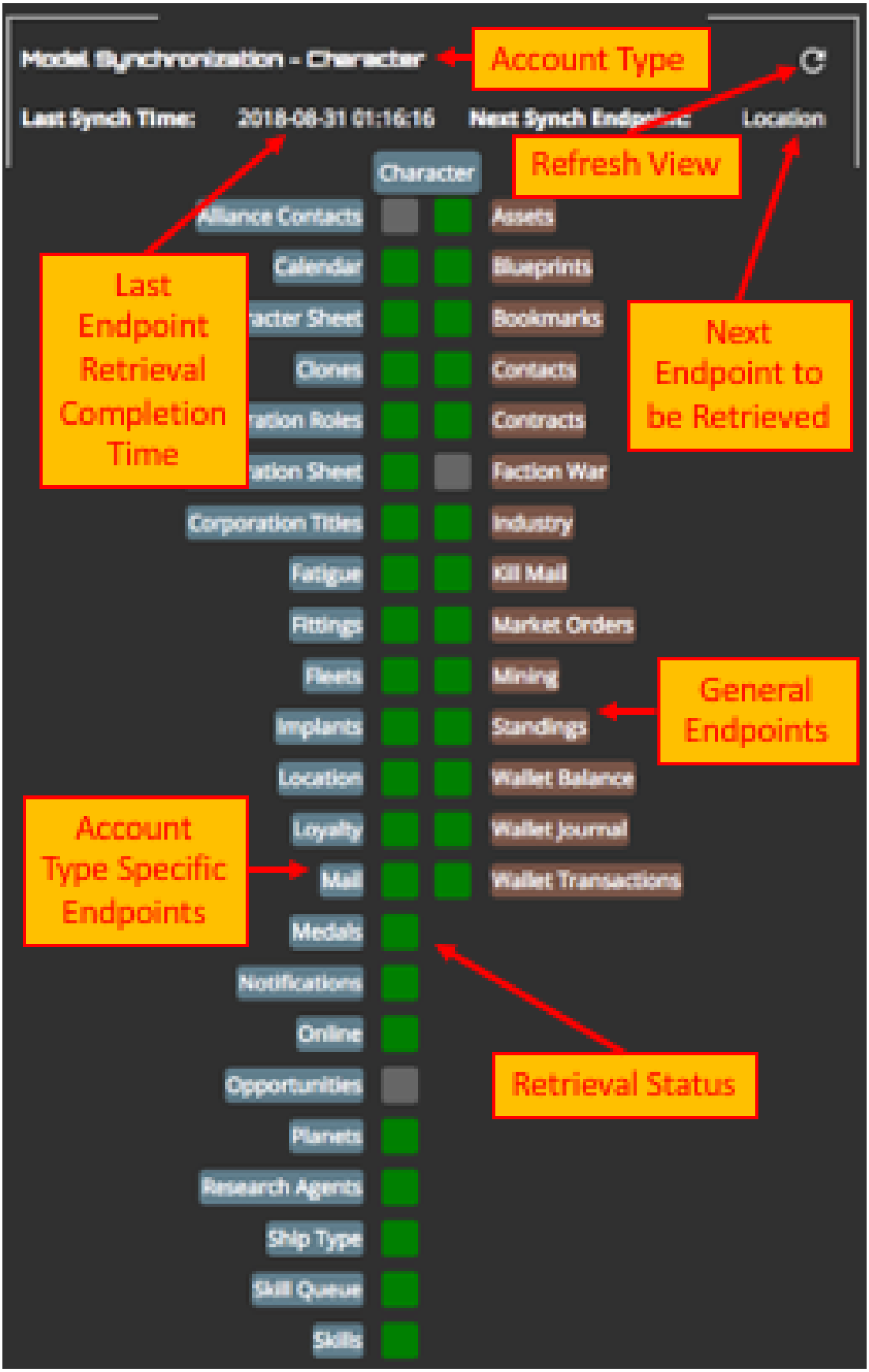
For each key, the key name is displayed followed by the key ID and hash, the expiry date of the key, and the key limit. The key ID and hash are needed by any application which wishes to use the key to access account data. The key expiry value gives the date after which the key will no longer function. The key limit value gives the date of the oldest account data accessible with this key. You can use key limits to prevent key users from looking too far in the past with respect to account data.

To the right of each key listing are a set of controls for listing key permissions (shield icon), using the key to explore account data (door icon), modifying key settings - including key name, expiry and limit dates, and all permissions (gear icon) - and finally, removing the key (garbage can icon).

If the 'explore account data' control is selected, then the view is changed to the model API screen pre-populated with the credentials for this key.

4.3 Synchronization Status

The last section in the account view shows the retrieval status (also called synchronization status) of each endpoint for the given account:


Account Type
Refresh View
Last Synch Time:
Location
Character
Alliance Contacts
Assets
Calendar
Blueprints
Bookmarks
Clones
Contacts
Contracts
Faction War
Industry
Corporation Titles
Kill Mail
Fatigue
Market Orders
Fittings
Mining
Fleets
Standings
Implants
Wallet Balance
Location
Wallet Journal
Loyalty
Wallet Transactions
Mail
Medals
Notifications
Online
Opportunities
Planets
Research Agents
Ship Type
Skill Queue
Skills
Last Endpoint Retrieval Completion Time
Next Endpoint to be Retrieved
General Endpoints
Account Type Specific Endpoints
Retrieval Status

The reload icon on the top right can be used to refresh status.

The center of this section indicates the account type, and therefore the set of endpoints which will be retrieved. The date and time of the last retrieval is recorded next to "Last Synch Time". The next endpoint which will be attempted is recorded next to "Next Synch Endpoint".

The left side of the display lists the endpoints specific to the account type. The right side of the display lists endpoints shared between character and corporation types.

Next to each endpoint is a status indicator showing the outcome of the most recent retrieval attempt. The status can be green (success), red (failure), yellow (timeout), or grey (not attempted - see below). Hovering over the status shows the time of the last attempt, and any detail regarding the status.

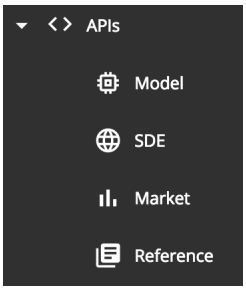
A grey status indicates one of the following conditions:

- The required scope is not included in the ESI token; or,
- Retrieval of the endpoint has been disabled by the administrator. This usually occurs either because of an EveKit bug (check the blog for status), or an ESI bug.

CHAPTER 5

API Screens

Most APIs supported by EveKit can be browsed from the API menu:



The API browser is a thin wrapper over the Swagger UI pointed at the appropriate API configuration file (usually a *swagger.json* file).

The model API provides access to your private character or corporation data, and therefore requires an EveKit access key. You can create an access key from the **'Account Screen'_** or by following the quickstart procedure (see **'Getting Started'_**). All other EveKit APIs are public and do not require credentials. The **'Other Datasets'_** section has additional information about the other data sets EveKit provides.

5.1 Creating an API Client

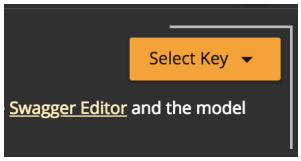
Each API is described by a Swagger configuration file. A link to the current Swagger configuration is displayed at the top of the API view for each section (e.g. `https://evekit-model.orbital.enterprises/api/swagger.json`). The easiest way to generate an API client is to use the online Swagger Editor. After loading this page, select "File -> Import URL" to import the appropriate Swagger configuration link. You can then use the "Generate Client" menu to generate a client in your favorite language.

If you'd prefer a more programmatic approach, you can use an online generator.

If you'd prefer even more control over the generation process (e.g. to provide support for a new language), you can clone the Swagger Codegen GitHub Project and build the generator yourself.

5.2 Model API

The model API is used to access your EVE Online character and corporation data. This API requires credentials in the form of an access key ID code and hash. Normally, you will supply these credentials as parameters to each API call. However, to make browsing easier, the model API view includes a key selector in the upper right corner:



Selecting a key using the key selector will pre-populate the appropriate credential fields in each model API call.

You can test an API call by selecting the 'Try it out' button for the appropriate call. All model API methods have default arguments which will retrieve the latest data for the selected endpoint. You can click the 'Execute' button to make a call. The result of the call (in JSON format) will be show in the 'Response body' section.

The model API is divided into five sections:

- **Access Key**: these endpoints provide information about a given model access key.
- **Character**: these endpoints provide access to character model data.
- **Corporation**: these endpoints provide access to corporation model data.
- **Common**: these endpoints provide access to data common to both characters and corporations.
- **Meta**: these endpoints allow for setting and retrieving meta data associated with model objects.

We describe various features of each section below.

5.2.1 Access Key Endpoints

The two access key endpoints are intended to be used by third party developers who wish to integrate EveKit into their applications. These endpoints can be used to discover certain properties of EveKit access keys such as:

- Whether this key provides access to character or corporation data;
- The names of the character and corporation associated with the key;
- The expiry time of the key (if any);
- The historic query limit of the key (if any);
- The access mask for this key; The mask determines which EveKit data may be access using this key; and,
- Whether the ESI token associated with the key is still valid; Third party applications may wish to use this information to alert their users when their EveKit data may have stoped updating due to a stale ESI token.

5.2.2 Model Data Endpoints

The model data endpoints - character, corporation and common - provide access to account model data. Each endpoint in this group has the following common argument list:

1. **accessKey (REQUIRED)**: The numeric id for the access key.
2. **accessCred (REQUIRED)**: The alphanumeric hash code for the access key.
3. **at (OPTIONAL)**: The model lifeline selector. This argument determines the date range of the model data to retrieve. If omitted, then the current date/time is used.

4. **contid (OPTIONAL)**: The continuation ID from which to start returning results. This argument sets the minimum (or maximum if **reverse** is true) "cached data" ID (**cid** - see **'EveKit Data Model'_**) which will be returned in the result set. This argument is used to page through large result sets (see below). If omitted, then results are returned starting from the first result (or last result if **reverse** is true) in the result set ordered by cid.
5. **maxresults (OPTIONAL)**: The maximum number of results to return. At time of writing, each endpoint will return no more than 1000 results. You must page through large result sets (see below) to retrieve all values. If omitted, then the maximum number of results allowed are returned.
6. **reverse (OPTIONAL)**: If true, then return results in descending order by cid. If omitted, then results are returned in ascending order by cid.
7. **endpoint specific arguments (OPTIONAL)**: Each endpoint may have zero or more additional arguments.

The access key and credential arguments are required for every call. The remaining arguments are optional and control the size and content of the result set. Each optional argument has a sensible default, but can also be modified as described below in order to alter the result set.

5.2.2.1 Point-in-Time and Historical Results

A key feature of EveKit is the retention of all history. To implement this feature, each model object is versioned and records a time range indicating when the model data was considered current. This time range is called a "lifeline" and is represented as a half-open interval *[s,e)*. Given a time, *t*, a model object is live at *t* if *s* <= *t* < *e*. The lifeline (**at**) argument allows you to specify *t*, and thus constrain the result set to only those model objects that were live at *t*.

The lifeline (**at**) argument supports a special query syntax as follows:

- **(omitted)**: if you do not specify a lifeline argument, then *t* is set to the current time. This is equivalent to `{values: [<current time>]}` as described below.
- **set**: the syntax `{values: [t1, t2, t3,...]}` will constrain the result set to model objects which were live at any of the times `t1`, `t2`, `t3`, etc. **NOTE: unless otherwise specified, all time arguments in EveKit are numeric values representing milliseconds since the epoch (January 1, 1970, UTC).**
- **range**: the syntax `{start: s, end: e}` will constrain the result set to model objects which were live at *any* time during the closed interval *[s, e]* (that is, inclusive).
- **all**: the syntax `{any: true}` will return all model objects, regardless of lifeline.

Specifying a large time range may require paging the result set as described below.

5.2.2.2 Result Set Filtering

It is often convenient to filter the result set according to model object properties. For example, we might wish to filter the `/ws/v1/common/blueprint` endpoint so that only bluesprints with a specific type ID are returned. To allow for model filtering, each endpoint includes an argument for each data field stored in a model object. These arguments can specify a filter using syntax similar to that described in the previous section:

- **(omitted)**: if you do not specify a filter for a model data field, then that field will not be used to constrain the result set. This is equivalent to the syntax `{any: true}` as described below.
- **set**: the syntax `{values: [v1, v2, v3,...]}` will constrain the result set to model objects where the given data field has at least one of the specified values. If the data field has type string, then the value arguments should be enclosed in double quotes (e.g. `{values: ["v1", "v2", "v3", ...]}`). Otherwise, the values will be interepreted as either boolean or numeric values as appropriate (see the note above concerning the specification of time arguments in EveKit).
- **range**: the syntax `{start: s, end: e}` will constrain the result set to the model objects where the given data field has a value in the specified range (inclusive). It is assumed that *s* <= *e* as determined by the

data field type (e.g. numerical ordering for numeric fields, lexicographic ordering for string fields, undefined for boolean fields).

- **string match**: the syntax `{like: m}` will constrain the result set to the model objects where the given data field is of string type and matches the wildcard expression `m`. In this case, `m` must be a quoted string in a format suitable for the SQL `LIKE` query. Such a string may be `%` to match any number of characters, and _ to match a single character. See SQL LIKE for more details.
- **all**: the syntax `{any: true}` which will return all model objects, regardless of the value of the given data field. Since this case is equivalent to specifying no filter, we only present it here for completeness.

The careful reader may notice the similarity to JSON syntax. This is no coincidence and, in fact, the syntax for both lifeline (**at**) and model data arguments is indeed just JSON in one of the formats specified above.

5.2.2.3 Paging Large Result Sets

At time of writing, the size of the result set is restricted to 1000 objects. For larger result sets, it is necessary to "page" through the results. This is accomplished by altering the **contid** argument on subsequent calls. The following python pseudo code illustrates this process:

```
contid = 0
# You can set reverse to true if you want descending results
reverse = false
results = []
next = call_endpoint(reverse=reverse, ...)
while len(next) > 0:
    results.extend(next)
    contid = next[-1].cid
    next = call_endpoint(contid=contid, reverse=reverse, ...)
```

This code takes advantage of the fact that results are always ordered by the "cached data" ID (cid). This order will be increasing if `reverse` is false, and decreasing otherwise. Therefore, the cid of the last object in the result list always gives the upper or lower bound (according to `reverse`) for the next results page.

5.2.3 Meta-Data Endpoints

Each model object supports meta-data in the form of a string-valued map (with string-valued keys). A special property of this map is that it is preserved when objects are evolved as part of data synchronization. This allows third party applications to store state tagged to model objects, and have that state preserved even as the object evolves over time. This process is described in more detail in EveKit Data Model.

The five meta-data endpoints allow you to read, set or remove meta-data attached to model objects. At time of writing, each object is limited to 10 meta-data entries.

5.3 Static Data Export (SDE) API

The Static Data Export (SDE) API provides online access to the latest Static Data Export released by CCP. EveKit provides access to the two most recent releases. You can select a release using the release selector in the upper right corner:

Selecting a release will cause the SDE API browser to appear.

The SDE API is divided into sections that mirror the tables provided by the CCP data file export. At time of writing, there were 15 sections. The SDE API endpoints are similar to the Model API endpoints and consist of the following argument list:

1. **contid (OPTIONAL)**: The continuation ID from which to start returning results. This argument represents an index into the result set, starting from 0. Use this argument to page through large result sets (see below).
2. **maxresults (OPTIONAL)**: The maximum number of results to return. At time of writing, each endpoint will return no more than 1000 results. You must page through large result sets (see below) to retrieve all values. If omitted, then the maximum number of results allowed are returned.
3. **endpoint specific arguments (OPTIONAL)**: Each endpoint may have zero or more additional arguments.

SDE API results can be filtered in the same way as described for the model API above. In particular, each endpoint specific argument accepts the same filtering syntax as above (e.g. set, range, or string match filtering).

The following python pseudo code illustrates the retrieval of large result sets:

```
contid = 0
results = []
next = call_endpoint(...)
while len(next) > 0:
    results.extend(next)
    contid += len(next)
    next = call_endpoint(contid=contid, ...)
```

Note that there is no concept of "reversing" the order of the result set. Results are always returned in the same order, but can be offset using the "contid" argument.

5.4 Market API

The Market API provides online access to regular snapshots of EVE Online market data, including order books and market history. The API consists of two components:

1. The *Online API*, described here, is a swagger annotated API much like the APIs described above. Use this API when you need to retrieve a small amount of data. For example, the order book for Tritanium in Jita an hour ago. Or, the last month of market history for Hulks in Amarr. The online API is *not* efficient for bulk or large volume retrieval.
2. The *Market Data Archive*, described in more detail in the *Market Data* section, stores daily order book and market history in a format convenient for bulk retrieval. Use the archive when you need larger data sets. For example, all market data for a given day across all EVE regions.

The online API has four simple endpoints:

1. **history**: retrieve market history for a given type, region and date.
2. **book**: retrieve the complete order book for a given type, region and date.
3. **livebook**: retrieve the latest order book for a given type and region.
4. **livestructure**: retrieve the latest order book for a given type and a limited set of player-owned structures. At time of writing, we record book data from the four highest market bolume structures, which are:
   (a) **1022409209010**: Hek - IChooseYou Trade Hub
   (b) **1023164547009**: Perimeter - - IChooseYou Trade Hub
   (c) **1023968078820**: Ashab - Port Royal - EVE-Mogul

(d) **1023075604524**: Perimeter - Planet V Panfam Secured Hub

This list is subject to change. Check the market data API page for the latest list.

Type and region arguments are the numeric IDs for these entities, which you can retrieve from the SDE (see previous section). Date arguments can either be milliseconds since the epoch (January 1, 1970 UTC) or any date parseable by the Javascript Date class.

Market history represents a daily summary and is normally not available until several hours after the end of the previous day. Order book data is snapped regularly throughout the day in five minute intervals. The snap frequency is set by the cache timers enforced by the EVE Swagger Interface (ESI).

Although we make it possible to get the latest order book data for some player-owned structures, we do not yet, at time of writing, process player-owned market data into files in the market data archive.

5.5 Reference API

The Reference API provides access to certain reference data provided by the ESI. Specifically:

- **alliances**: these endpoints provide access to the list of alliances and their membership.
- **faction**: these endpoints provide access to faction warfare statistics.
- **server**: this endpoint provides access to server status.
- **sovereignty**: these endpoints provide access to the sovereignty map, structures, and campaigns.

While the ESI already provides live access to this data, EveKit goes one step further and retains a history for this data, similar to the way history is retained for character and corporation data.

The Reference API is public and does not require credentials for access. Endpoints are structured similar to the model API with the exclusion of credential arguments:

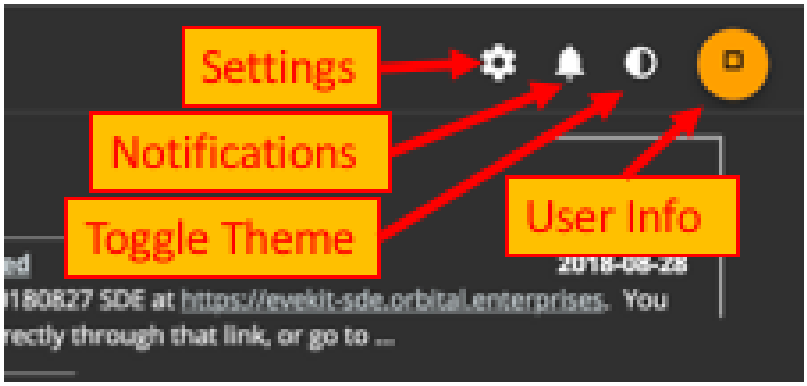
1. **at (OPTIONAL)**: The lifeline selector. This argument determines the date range of the reference data to retrieve. If omitted, then the current date/time is used.
2. **contid (OPTIONAL)**: The continuation ID from which to start returning results. This argument sets the minimum (or maximum if **reverse** is true) "cached data" ID (**cid** - see **'EveKit Data Model'_**) which will be returned in the result set. This argument is used to page through large result sets (see below). If omitted, then results are returned starting from the first result (or last result if **reverse** is true) in the result set ordered by cid.
3. **maxresults (OPTIONAL)**: The maximum number of results to return. At time of writing, each endpoint will return no more than 1000 results. You must page through large result sets to retrieve all values. If omitted, then the maximum number of results allowed are returned.
4. **reverse (OPTIONAL)**: If true, then return results in descending order by cid. If omitted, then results are returned in ascending order by cid.
5. **endpoint specific arguments (OPTIONAL)**: Each endpoint may have zero or more additional arguments.

Point-in-time and historical queries are accomplished with the same syntax presented in *Point-in-Time and Historical Results*. Likewise, endpoint specific arguments may be used as filters as described in *Result Set Filtering*. Finally, large result sets can be paged using the technique described in *Paging Large Result Sets*.

CHAPTER 6

Settings, Notifications, and User Information

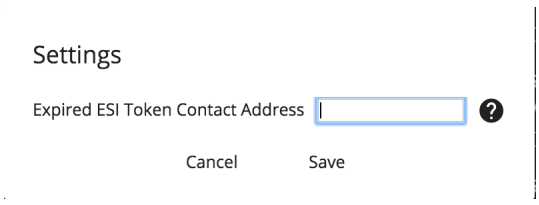
The upper right portion of the display contains the EveKit toolbar:



The toolbar has four elements:

- **settings**: common settings shared across all synchronized accounts.
- **notifications**: notifications about various account activities.
- **theme**: a toggle to switch between light and dark themes.
- **user information**: information about the logged in user.

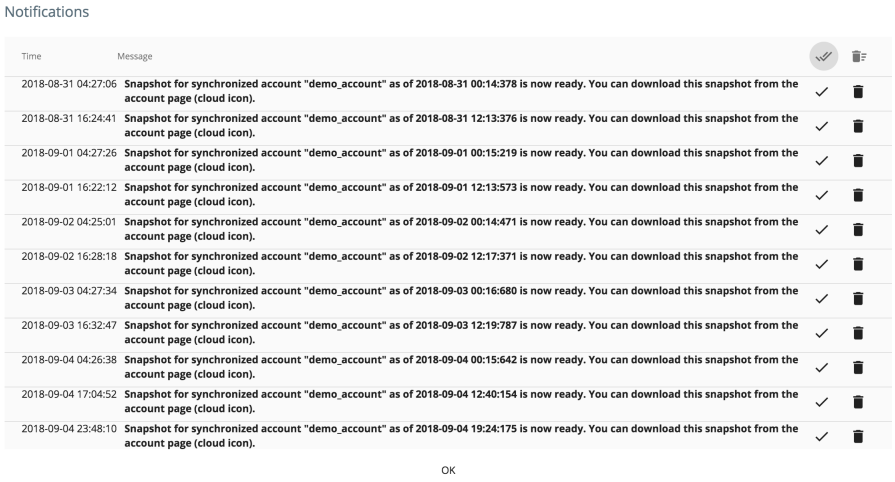
Clicking on the settings button will display the settings dialog:



This dialog will display various options which allow you to customize EveKit according to your needs. At time of writing, the only setting is "Expired ESI Token Contact Address" which allows you to set an e-mail address which is

alerted if one of your ESI tokens can not be refreshed.

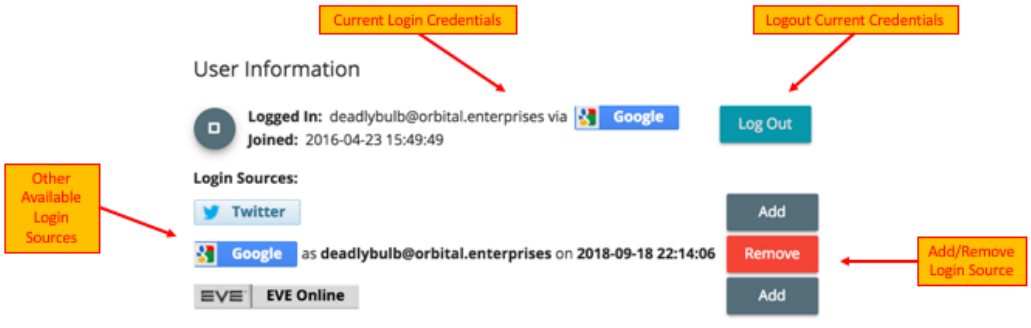
Clicking on the notifications button will display the notifications dialog:



Notifications are typically informational events about one or more of your synchronized accounts. At time of writing, a notification is generated each time an account snapshot is generated (normally once a day). Unread notifications will cause the notifications button to be decorated with a different color and the count of unread notifications. You can read and/or delete notifications at your convenience. Note, however, that notifications are automatically deleted after 30 days (read or otherwise).

Clicking on the theme button will toggle between light and dark themes. Theme preferences are saved in a cookie on your browser so that they are preserved between sessions.

Finally, clicking the user information button will display the user info dialog:



The user information dialog shows the credentials used for the current login along with the date when this user joined EveKit. Next to the current credentials is the "logout" button which will immediately logout the current user.

EveKit supports multiple login sources although, typically, users will only use one source at a time. If you need to use multiple sources or, more commonly, you have accidentally used multiple sources, then the user information dialog

will let you manage your login sources as follows:

- *If you wish to use multiple sources...* login with your prefered source and pull up the user information dialog. In the "login sources" section, click the "add" button of the additional source you would like to use. This will immediately start the authentication process for the chosen source. When this process completes, you will be logged in with the new source and all your sources will be tied together. You can once again pull up the user information dialog to verify all of your desired sources have been added.
- *If you accidentially logged in with a different source...* you can always just logout and log back in with your preferred source. However, if you'd like to combine your sources (or remove the other source you used), then:

  1. Login with your preferred source.
  2. Pull up the user information dialog.
  3. In the "login sources" section, click the "add" button on the other source you used previously.

  This will immediately start the authentication process for the added source. When this process completes, you will be logged in with the new source and all your sources will be tied together (which you can verify by pulling up the user information dialog). If you wish, you can now remove any sources you don't want as described below.
- *If you wish to delete a source...* login with any source you have used before and pull up the user information dialog. In the "login sources" section, click the "remove" button of the source you would like to remove. The selected source will be immediately removed. If the selected source happens to be your current login source, then you will also be logged out. If you only have one login source, then the UI will normally try to prevent you from removing your only source. If you manage to somehow remove your only source, you will need to contact the administrator to restore access to your account.

CHAPTER 7

EveKit Data Model

EVE data stored by EveKit is recorded in "Cached Data" objects. These objects model the data retrieved from the ESI. Every such object has the following common properties:

- **owner**: the synchronized EVE account to which the object is attached.
- **cached object id (cid)**: the globally unique ID of this object.
- **life start**: the time (milliseconds UTC) at which the object was considered current for the data it models.
- **life end**: the time (milliseconds UTC) at which the object was considered no longer current for the data it models.
- **meta data**: a string map (keys are also strings) which may store additional properties associated with the object.

Each object may then have additional properties depending on the data it is intended to model. For example the `AccountBalance` object has the following additional properties:

- **division**: the wallet division for which the balance is recorded.
- **balance**: the current account balance.

A model object is created when it becomes current for the data it models. At this point, all fields are considered immutable except for "life end" and the meta-data map. The "life end" field may be modified during object history maintenance. The meta-data map may be modified at any time. "Life end" is normally set only once at the time when the object is no longer current. After that point, all model object properties except the meta-data map are considered immutable.

7.1 Model History

A key feature of EveKit is the ability to record history for the data retrieved from the ESI. History is maintained by creating cached data object instances which represent a view of data over a given time interval. For a given object, the half-open interval [life start, life end) is the time range over which the object is considered to be authoritative for the data it models. Outside of this interval, the modeled data either didn't exist, or had different values for one or more object properties.

History is updated during each model synchronization cycle (see :ref:model_sync) as follows:

1. The latest ESI data is retrieved and a new model object is created. Life start is set to the current time, and life end is set to positive infinity.
2. If no model object exists for the current time, then the new model object is stored as the current view.
3. If a model object already exists for the current time, then it is retrieved and compared to the new model object:
   (a) if all fields are identical (excluding life start, life end and meta-data) then nothing has changed and the new model object is discarded.
   (b) if a field has changed, then:
      i. life end for the current object is set to the current time.
      ii. the meta-data map for the existing object is copied to the new object.
      iii. both objects are saved, with the new object becoming the current view and the old object becoming a historical view.

Step 3.b. is called "object evolution" and preserves the property that [life start, life end) intervals are non overlapping for all objects for a particular model type (e.g. all `AccountBalance` model objects).

One may wonder why it is necessary to copy the meta-data map during object evolution. This is necessary to preserve third party data for model objects which regularly evolve. For example, consider a character sheet. A player's character sheet naturally evolves over time as security status changes, or a player updates their description. A third party application may choose to save certain preferences in the meta-data map of the character sheet. By copying this map during evolution, these preferences continue to be convenient to access on current view of the character sheet. Note that historical object still have a meta-data map which can be accessed or modified independent of the latest model view.

7.2 Model Data Reference

7.2.1 `AccountBalance`

Category	Shared	
Access	Account Balance Access (ACCESS_ACCOUNT_BALANCE)	
Field Name	**Type**	**Description**
`division`	number	Wallet division ('1' for character accounts).
`balance`	currency	Wallet balance.
`accountID`	number	(deprecated) Always 0.
`accountKey`	number	(deprecated) Set to *division - 1 + 1000*.

7.2.2 Asset

Category	Shared	
Access	Asset Access (ACCESS_ASSETS)	
Field Name	**Type**	**Description**
itemID	number	Unique item ID.
locationID	number	Either an EVE location (see SDE) or the itemID of the parent if this is a contained asset.
locationType	string	One of “station”, “solar sytem”, or “other”.
locationFlag	string	Location descriptor (see location_flag).
typeID	number	Asset type ID.
quantity	number	Asset stack count.
singleton	boolean	True if this asset is not a stack.
blueprintType	string	(deprecated) Records blueprint informaion for pre-ESI data. One of “copy”, “original” or null.
blueprintCopy	boolean	True if this asset represents a blueprint copy.

7.2.3 Blueprint

Category	Shared	
Access	Blueprint Access (ACCESS_BLUEPRINTS)	
Field Name	**Type**	**Description**
itemID	number	Unique item ID.
locationID	number	EVE location (see SDE).
locationFlag	string	Location descriptor (see location_flag).
typeID	number	Blueprint type ID.
quantity	number	Blueprint stack count (see quantity).
timeEfficiency	number	Blueprint time efficiency.
materialEfficiency	number	Blueprint material efficiency.
runs	number	Runs remaining for a blueprint copy.

7.2.4 Bookmark

Category	Shared	
Access	Bookmark Access (ACCESS_BOOKMARKS)	
Field Name	**Type**	**Description**
folderID	number	Bookmark folder ID where this bookmark is stored. In EveKit, bookmark folders are not stored separately and are recorded along with the bookmarks they contain. As a result, empty bookmark folders are not explicitly recorded.
folderName	string	Bookmark folder name where this bookmark is stored.
folderCreatorID	number	Bookmark folder creator ID. Only stored for corporation bookmarks.
bookmarkID	number	Unique bookmark ID.
bookmarkCreatorID	number	Bookmark creator ID.
created	number	Bookmark creation time (millis UTC).
createdDateStr	string	Bookmark creation time (YYYY-MM-DDTHH:MM:SS.sssZ).
itemID	number	Item ID bookmark refers to or 0 if not present.
typeID	number	Type ID of item bookmark refers to or 0 if not present.
locationID	number	Bookmark location ID.
x	number	Bookmark X position or 0 if not present.
y	number	Bookmark Y position or 0 if not present.
z	number	Bookmark Z position or 0 if not present.
memo	string	Bookmark label. The XML API called this field memo. We've retained the name but record label here instead.
note	string	Boomkark note

7.2.5 Contact

Category	Shared	
Access	Contact List Access (ACCESS_CONTACT_LIST)	
Field Name	**Type**	**Description**
list	string	Which contact list this contact is in. One of "character", "corporation" or "alliance".
contactID	number	Unique contact ID.
standing	number	Contact standing.
contactType	string	One of "character", "corporation", "alliance", or "faction".
inWatchList	boolean	True if this contact is in the watched list.
isBlocked	boolean	True if this contact is in the blocked list.
labels	set<nunmber>	Contact label IDs attached to this contact.

7.2.6 ContactLabel

Category	Shared	
Access	Contact List Access (ACCESS_CONTACT_LIST)	
Field Name	**Type**	**Description**
list	string	Which contact list this contact label is in. One of "character", "corporation" or "alliance".
labelID	number	Unique label ID.
name	string	Label name.

7.2.7 Contract

Category	Shared	
Access	Contract Access (ACCESS_CONTRACTS)	
Field Name	**Type**	**Description**
contractID	number	Unique contract ID.
issuerID	number	Contract issuer ID.
issuerCorpID	number	Corporation ID of contract issuer.
assigneeID	number	Contract assignee ID (if assigned).
acceptorID	number	Contract acceptor ID (if accepted).
startStationID	number	Starting station ID (for courier contracts).
endStationID	number	Ending station ID (for courier contracts).
type	string	Contract type. One of “item_exchange”, “auction”, “courier”, “loan”, or “unknown”.
status	string	Contract status (see status).
title	string	Contract title.
forCorp	boolean	True if the contract was issued on behalf of the issuer’s corporation.
availability	string	Contract availability. One of “public”, “personal”, “corporation” or “alliance”.
dateIssued	number	Date of contract issue (millis UTC).
dateIssuedDate	string	Date of contract issue (YYYY-MM-DDTHH:MM:SS.sssZ).
dateExpired	number	Date contract expires (millis UTC).
dateExpiredDate	string	Date contract expires (YYYY-MM-DDTHH:MM:SS.sssZ).
dateAccepted	number	Date contract was accepted (millis UTC) or 0 if not accepted yet.
dateAcceptedDate	string	Date contract was accepted (YYYY-MM-DDTHH:MM:SS.sssZ).
numDays	number	Number of days allocated to complete the contract.
dateCompleted	number	Date contract completed (millis UTC) or 0 if not completed yet.
dateCompletedDate	string	Date contract completed (YYYY-MM-DDTHH:MM:SS.sssZ).
price	currency	Contract price.
reward	currency	Contract reward.
collateral	currency	Contract collateral.
buyout	currency	Contract buyout (for auctions).
volume	number	Volume of items in the contract.

7.2.8 ContractBid

Category	Shared	
Access	Contract Access (ACCESS_CONTRACTS)	
Field Name	**Type**	**Description**
bidID	number	Unique (to contract) bid ID.
contractID	number	Unique contract ID.
bidderID	number	Contract bidder ID.
dateBid	number	Date of bid (millis UTC).
dateBidDate	string	Date of bid (YYYY-MM-DDTHH:MM:SS.sssZ).
amount	currency	Bid amount.

7.2.9 ContractItem

Category	Shared	
Access	Contract Access (ACCESS_CONTRACTS)	
Field Name	**Type**	**Description**
contractID	number	Unique contract ID.
recordID	number	Unique item ID.
typeID	number	Type ID of item.
quantity	number	Item count.
rawQuantity	number	-1 for a non-stackable item, or a blueprint original. -2 for a blueprint.
singleton	boolean	True if item is a singleton.
included	boolean	True if this item is offered as part of the contract. False if this item is being asked for in the contract.

7.2.10 FacWarStats

Category	Shared	
Access	Faction War Stats Access (ACCESS_FAC_WAR_STATS)	
Field Name	**Type**	**Description**
currentRank	number	Current faction war rank.
enlisted	number	Enlistment date (millis UTC).
enlistedDate	string	Enlistment date (YYYY-MM-DDTHH:MM:SS.sssZ).
factionID	number	Faction ID.
highestRank	number	Highest rank achieved in faction.
killsLastWeek	number	Kills last week.
killsTotal	number	Total kills.
killsYesterday	number	Kills yesterday.
pilots	number	Pilot count.
victoryPointsLastWeek	number	Victory points earned last week.
victoryPointsTotal	boolean	Total victory points earned.
victoryPointsYesterday	boolean	Victory points earned yesterday.

7.2.11 IndustryJob

Category	Shared	
Access	Industry Jobs Access (ACCESS_INDUSTRY_JOBS)	
Field Name	**Type**	**Description**
jobID	number	Unique job ID.
installerID	number	Job installer ID.
facilityID	number	Facility ID where job is installed.
stationID	number	Station or location ID where job facility is located.
activityID	number	Job activity ID.
blueprintID	number	Blueprint ID used for job.
blueprintTypeID	number	Blueprint type ID used for job.
blueprintLocationID	number	Station or location ID where blueprint is located.
outputLocationID	number	Station or location ID where job output will be placed.
runs	number	Job runs.
cost	currency	Job cost.
licensedRuns	number	Number of runs blueprint is licensed for.
probability	number	Probability of job success (invention jobs only).
productTypeID	number	Type ID of product manufactured, copied, or invented.
status	string	Job status. One of “active”, “cancelled”, “delivered”, “paused”, “ready” or “reverted”.
timeInSeconds	number	Job duration in seconds. ESI now calls this “duration”.
startDate	number	Job start date (millis UTC).
startDateDate	string	Job start date (YYYY-MM-DDTHH:MM:SS.sssZ).
endDate	number	Job end date (millis UTC).
endDateDate	string	Job end date (YYYY-MM-DDTHH:MM:SS.sssZ).
pauseDate	number	Job pause date (millis UTC) or 0 if not paused.
pauseDateDate	string	Job pause date (YYYY-MM-DDTHH:MM:SS.sssZ).
completedDate	number	Job completed date (millis UTC) or 0 if not completed.
completedDateDate	string	Job completed date (YYYY-MM-DDTHH:MM:SS.sssZ).
completedCharacterID	number	Character ID which completed the job.
successfulRuns	number	Number of successful job runs.

7.2.12 Kill

Category	Shared	
Access	Kill Log Access (ACCESS_KILL_LOG)	
Field Name	**Type**	**Description**
killID	number	Unique kill ID.
killTime	number	Kill time (millis UTC).
killTimeDate	string	Kill time (YYYY-MM-DDTHH:MM:SS.sssZ).
moonID	number	Moon ID if the kill took place at a moon.
solarSystemID	number	Solar system ID where the kill took place.
warID	number	War ID if the kill was generated as part of an official war.

7.2.13 KillAttacker

Category	Shared	
Access	Kill Log Access (ACCESS_KILL_LOG)	
Field Name	**Type**	**Description**
killID	number	Unique kill ID.
attackerCharacterID	number	Attacker's character ID.
allianceID	number	Attacker alliance ID, or 0 if attacker not in an alliance.
attackerCorporationID	number	Attacker's corporation ID.
damageDone	number	Damage done by attacker.
factionID	number	Attacker faction ID, or 0 if attacker not in a faction.
securityStatus	number	Attacker's security status.
shipTypeID	number	Attacker's ship type ID.
weaponTypeID	number	Attacker's weapon type ID.
finalBlow	boolean	True if this attacker inflicted the final blow.

7.2.14 KillItem

Category	Shared	
Access	Kill Log Access (ACCESS_KILL_LOG)	
Field Name	**Type**	**Description**
killID	number	Unique kill ID.
typeID	number	Item type ID.
flag	number	Item location flag.
qtyDestroyed	number	Quantity of item destroyed.
qtyDropped	number	Quantity of item dropped.
singleton	number	Singleton flag.
sequence	number	Synthetic field introduced by EveKit. This field enumerates kill items to differentiate items with the same type ID.
containerSequence	number	Synthetic field introduced by EveKit. This field gives the sequence number of the container for an item (if it had one), or -1 if an item had no container.

7.2.15 KillVictim

Category	Shared	
Access	Kill Log Access (ACCESS_KILL_LOG)	
Field Name	**Type**	**Description**
killID	number	Unique kill ID.
allianceID	number	Alliance ID of victim, or 0 if not in an alliance.
killCharacterID	number	Victim character ID .
killCorporationID	number	Victim corporation ID.
damageTaken	number	Victim damage taken.
factionID	number	Faction ID of victim, or 0 if not in a faction.
shipTypeID	number	Victim ship type ID.
x	number	Victim's x position at time of kill.
y	number	Victim's y position at time of kill.
z	number	Victim's z position at time of kill.

7.2.16 Location

Category	Shared	
Access	Locations Access (ACCESS_LOCATIONS)	
Field Name	**Type**	**Description**
itemID	number	Item ID.
itemName	string	Item name.
x	number	Item x position.
y	number	Item y position.
z	number	Item z position.

7.2.17 MarketOrder

Category	Shared	
Access	Market Orders Access (ACCESS_MARKET_ORDERS)	
Field Name	**Type**	**Description**
orderID	number	Unique order ID.
walletDivision	number	Wallet division from which order was placed. Always 1 for character orders.
bid	boolean	True for bid (buy) orders, false for ask (sell) orders.
charID	number	(deprecated) Character ID which issued the order. Only present on pre-ESI orders.
duration	number	Number of days for which the order is valid, starting from the issued date.
escrow	currency	The amount of ISK in escrow for buy orders.
issued	number	Order issue date (millis UTC).
issuedDate	string	Order issue date (YYYY-MM-DDTHH:MM:SS.sssZ).
issuedBy	number	ID of character who issued this order. Only present for corporation orders.
minVolume	number	Minimum acceptable quantity for buy orders.
orderState	string	One of “cancelled”, “expired” or “open”. The “open” state is introduced by EveKit to distinguish live orders.
price	currency	Order price.
orderRange	string	Order range in number of jumps. May be one of “1”, “2”, “3”, “4”, “5”, “10”, “20”, “30”, “40”, “region”, “solarsystem” or “station”.
typeID	number	Type ID of item transacted in this order.
volEntered	number	Volume entered at order creation.
volRemaining	number	Volume remaining for order.
regionID	number	Region ID where order was placed.
locationID	number	Location ID where order was placed.
isCorp	boolean	If true, then this order was placed on behalf of the placing character’s corporation.
accountKey	number	(deprecated) Set to *walletDivision -1 + 1000*.

7.2.18 Standing

Category	Shared	
Access	Standings Access (ACCESS_STANDINGS)	
Field Name	**Type**	**Description**
standingEntity	string	One of “agent”, “npc_corp” or “faction”.
fromID	number	ID of entity for standing.
standing	number	Entity standing.

7.2.19 WalletJournal

Category	Shared	
Access	Wallet Journal Access (ACCESS_WALLET_JOURNAL)	
Field Name	**Type**	**Description**
division	number	Wallet division. 1 for charactre wallet journal entries.
refID	number	Unique journal reference ID.
date	number	Date of entry (millis UTC).
dateDate	string	Date of entry (YYYY-MM-DDTHH:MM:SS.sssZ).
refType	string	Transaction type. See ref_type.
firstPartyID	number	The ID of the first party involved in the transaction.
secondPartyID	number	The ID of the second party involved in the transaction.
argName1	string	(deprecated) Only present on historic data. null for ESI data.
argID1	number	(deprecated) Only present on historic data. 0 for ESI data.
amount	currency	Transaction amount. Positive for ISK deposited, negative for ISK withdrawn.
balance	currency	Wallet balance after transaction completed.
reason	string	User-provided reason for transaction.
taxReceiverID	number	The corporation ID receiving any tax paid.
taxAmount	currency	The amount of tax paid.
contextID	number	An ID related to the context as determined by *refType*.
contextType	string	Type of *contextID* if present. See context_id_type.
description	string	Reason for the transaction as shown in the client.
accountKey	number	(deprecated) Set to *division - 1 + 1000*.
ownerID1	number	(deprecated) Set to *firstPartyID*.
ownerID2	number	(deprecated)Set to *secondPartyID*.

7.2.20 WalletTransaction

Category	Shared	
Access	Wallet Transactions Access (ACCESS_WALLET_TRANSACTIONS)	
Field Name	**Type**	**Description**
division	number	Transaction division. 1 for character transactions.
transactionID	number	Unique transaction ID.
date	number	Transaction date (millis UTC).
dateDate	string	Transaction date (YYYY-MM-DDTHH:MM:SS.sssZ).
quantity	number	Number of items transacted.
typeID	number	Type ID of item transacted.
price	currency	Price for each item.
clientID	number	ID of client with which transaction occurred.
locationID	number	Location where transaction occurred.
isBuy	boolean	True for a buy, false otherwise.
isPersonal	boolean	True if on behalf of the transacting character, false if on behalf of the character's corporation.
journalTransactionID	number	Corresponding journal reference ID. -1 if no such entry exists.
accountKey	number	(deprecated) Set to *division - 1 + 1000*.
stationID	number	(deprecated) Set to *locationID*.
transactionType	string	(deprecated) Set to "buy" if *isBuy = true*, otherwise set to "sell".
transactionFor	string	(deprecated) Set to "personal" if *isPersonal = true*, otherwise set to "corporation".

7.2.21 CalendarEventAttendee

Category	Character	
Access	Calendar Event Attende Access (ACCESS_CALENDAR_EVENT_ATTENDEES)	
Field Name	**Type**	**Description**
eventID	number	Unique calendar event ID.
characterID	number	ID of character responding.
response	string	Character response. One of "declined", "not_responded", "accepted" or "tentative".

7.2.22 CharacterContactNotification

Category	Character	
Access	Contact Notifications Access (ACCESS_CONTACT_NOTIFICATIONS)	
Field Name	**Type**	**Description**
notificationID	number	Unique notification ID.
senderID	number	Character ID of contact list this character has been added to.
sentDate	number	Notification send date (millis UTC).
sentDateDate	string	Notification send date (YYYY-MM-DDTHH:MM:SS.sssZ).
standingLevel	number	Standing tier this character has been assigned to in the adding character's list. See standing_level.
messageData	string	Notification message.

7.2.23 CharacterFleet

Category	Character	
Access	Charadcter Fleets Access (ACCESS_CHARACTER_FLEETS)	
Field Name	**Type**	**Description**
fleetID	number	Unique fleet ID.
role	string	Fleet role. One of "fleet_commander", "squad_commander", "squad_member" or "wing_commander".
squadID	number	Squad within fleet to which the character has been assigned, or -1 if not in a squad.
wingID	number	Wing within squad to which the character has been assigned, or -1 if not in a wing.

7.2.24 CharacterLocation

Category	Character	
Access	Locations Access (ACCESS_LOCATIONS)	
Field Name	**Type**	**Description**
solarSystemID	number	Solar system ID where character is located.
stationID	number	Station ID where character is located, or 0 if not located in a station.
structureID	number	Structure ID where character is located, or 0 if not located in a structure.

7.2.25 CharacterMailMessage

Category	Character	
Access	Mail Access (ACCESS_MAIL)	
Field Name	**Type**	**Description**
messageID	number	Unique message ID.
senderID	number	Character ID of sender.
sentDate	number	Message send date (millis UTC).
sentDateDate	string	Message send date (YYYY-MM-DDTHH:MM:SS.sssZ).
title	string	Message title.
msgRead	boolean	True if the message has been read. False otherwise.
labels	set<number>	Set of label IDs applied to message (see *MailLabel*).
recipients	set<*MailMessageRecipient*>	Set of message recipients.
body	string	Message contents.

7.2.26 CharacterMedal

Category	Character	
Access	Medals Access (ACCESS_MEDALS)	
Field Name	**Type**	**Description**
description	string	Description of medal.
medalID	number	Unique medal ID.
title	string	Medal title.
corporationID	number	ID of corporation which issued medal.
issued	number	Medal issue date (millis UTC).
issuedDate	string	Medal issue date (YYYY-MM-DDTHH:MM:SS.sssZ).
issuerID	number	ID of character which issued medal.
reason	string	Reason medal was issued.
status	string	One of "public" or "private".

7.2.27 CharacterMedalGraphic

Category	Character	
Access	Medals Access (ACCESS_MEDALS)	
Field Name	**Type**	**Description**
medalID	number	Unique medal ID.
issued	number	Medal issue date (millis UTC).
issuedDate	string	Medal issue date (YYYY-MM-DDTHH:MM:SS.sssZ).
part	number	Part code.
layer	number	Layer code.
graphic	string	Graphic image name.
color	number	Color code, or 0 if none.

7.2.28 CharacterNotification

Category	Character	
Access	Notifications Access (ACCESS_NOTIFICATIONS)	
Field Name	**Type**	**Description**
notificationID	number	Unique notification ID.
type	string	Notification type. See type.
senderID	number	ID of sender. Interpretation depends on "senderType".
senderType	string	Sender type. One of "character", "corporation", "alliance", "faction" or "other".
sentDate	number	Notification send date (millis UTC).
sentDateDate	string	Notification send date (YYYY-MM-DDTHH:MM:SS.sssZ).
msgRead	boolean	True if the notification has been read, false otherwise.
text	string	Notification text.

7.2.29 CharacterOnline

Category	Character	
Access	Account Status Access (ACCESS_ACCOUNT_STATUS)	
Field Name	**Type**	**Description**
online	boolean	True if character online, false otherwise.
lastLogin	number	Last login time (millis UTC).
lastLoginDate	string	Last login time (YYYY-MM-DDTHH:MM:SS.sssZ).
lastLogout	number	Last logout time (millis UTC).
lastLogoutDate	string	Last logout time (YYYY-MM-DDTHH:MM:SS.sssZ).
logins	number	Total number of logins.

7.2.30 CharacterRole

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
roleCategory	string	Role category. This is a synthetic field created by EveKit to organize roles. Value will be one of "CORPORATION", "CORPORATION_AT_HQ", "CORPORATION_AT_BASE" or "CORPORATION_AT_OTHER".
roleName	string	Role name.

7.2.31 CharacterSheet

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
characterID	number	Unique character ID.
name	string	Character name (also unique).
corporationID	number	Character corporation ID.
raceID	number	Character race ID.
doB	number	Character date of birth (millis UTC).
doBDate	string	Character date of birth (YYYY-MM-DDTHH:MM:SS.sssZ).
bloodlineID	number	Character bloodline ID.
ancestryID	number	Character ancestry ID, or 0 if none.
gender	string	Character gender. Either "male" or "female".
allianceID	number	Character alliance ID, or 0 if none.
factionID	number	Character faction ID, or 0 if none.
description	string	Character provided description.
securityStatus	number	Character security status.

7.2.32 CharacterSheetAttributes

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
intelligence	number	Character intelligence.
memory	number	Character memory.
charisma	number	Character charisma.
perception	number	Character perception.
willpower	number	Character willpower.
bonusRemaps	number	Number of available bonus character neural remaps.
lastRemapDate	number	Date of last neural remap, including bonus remaps (millis UTC).
lastRemapDateDate	string	Date of last neural remap, including bonus remaps (YYYY-MM-DDTHH:MM:SS.sssZ).
accruedRemapCooldownDate	number	Date when neural remap cooldown will complete after using an accrued neural remap (millis UTC).
accruedRemapCooldownDateDate	string	Date when neural remap cooldown will complete after using an accrued neural remap (YYYY-MM-DDTHH:MM:SS.sssZ).

7.2.33 CharacterSheetClone

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
cloneJumpDate	number	Date of last clone jump (millis UTC).
cloneJumpDateDate	string	Date of last clone jump (YYYY-MM-DDTHH:MM:SS.sssZ).
homeStationID	number	Home station ID.
homeStationType	string	Home station type. One of “station” or “structure”.
lastStationChangeDate	number	Date of last home station change (millis UTC).
lastStationChangeDateDate	string	Date of last home station change (YYYY-MM-DDTHH:MM:SS.sssZ).

7.2.34 CharacterSheetJump

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
jumpActivation	number	Date of last jump activation (millis UTC).
jumpActivationDate	string	Date of last jump activation (YYYY-MM-DDTHH:MM:SS.sssZ).
jumpFatigue	number	Date of jump fatigue expiry (millis UTC).
jumpFatigueDate	string	Date of jump fatigue expiry (YYYY-MM-DDTHH:MM:SS.sssZ).
jumpLastUpdate	number	Date of last jump update (millis UTC).
jumpLastUpdateDate	string	Date of last jump update (YYYY-MM-DDTHH:MM:SS.sssZ).

7.2.35 CharacterSheetSkillPoints

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
totalSkillPoints	number	Total character skill points.
unallocatedSkillPoints	number	Unallocated skill points.

7.2.36 CharacterShip

Category	Character	
Access	Locations Access (ACCESS_LOCATIONS)	
Field Name	**Type**	**Description**
shipTypeID	number	Character's ship type ID.
shipItemID	number	Unique item ID identifying character's ship.
shipName	string	Ship name.

7.2.37 CharacterSkill

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
typeID	number	Skill type ID.
trainedSkillLevel	number	Maximum level trained in skill.
skillPoints	number	Skill points invested in skill.
activeSkillLevel	number	Current active skill level (may be reduced due to alpha clone status).

7.2.38 CharacterTitle

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
titleID	number	Unique title ID.
titleName	string	Title name.

7.2.39 `ChatChannel` (deprecated)

Category	Character	
Access	Chat Channels Access (ACCESS_CHAT_CHANNELS)	
Field Name	**Type**	**Description**
`channelID`	number	Unique channel ID.
`ownerID`	number	Character ID of channel creator.
`displayName`	string	Displayed name of channel.
`comparisonKey`	string	Key used to disambiguate chat channels.
`hasPassword`	boolean	True if the channel requires a password for entry, false otherwise.
`motd`	string	Message of the day.

7.2.40 `ChatChannelMember` (deprecated)

Category	Character	
Access	Chat Channels Access (ACCESS_CHAT_CHANNELS)	
Field Name	**Type**	**Description**
`channelID`	number	Unique channel ID.
`category`	string	One of “allowed”, “blocked”, “muted” or “operator”.
`accessorID`	number	ID of channel member.
`accessorType`	string	Channel member type.
`untilWhen`	number	If blocked, date at which block expires (millis UTC).
`untilWhenDate`	string	If blocked, date at which block expires (YYYY-MM-DDTHH:MM:SS.sssZ).
`reason`	string	Reason for blocking this member.

7.2.41 `Fitting`

Category	Character	
Access	Ship Fittings Access (ACCESS_FITTINGS)	
Field Name	**Type**	**Description**
`fittingID`	number	Unique fitting ID.
`name`	string	Fitting name.
`description`	string	Fitting description.
`shipTypeID`	number	Ship type ID described by fitting.

7.2.42 `FittingItem`

Category	Character	
Access	Ship Fittings Access (ACCESS_FITTINGS)	
Field Name	**Type**	**Description**
`fittingID`	number	Unique fitting ID.
`typeID`	number	Fitting type ID.
`flag`	number	Fitting flag.
`quantity`	number	Number of this item fit.

7.2.43 FleetInfo

Category	Character	
Access	Character Fleets Access (ACCESS_CHARACTER_FLEETS)	
Field Name	**Type**	**Description**
fleetID	number	Unique fleet ID.
isFreeMove	boolean	True if free movement is allowed, false otherwise.
isRegistered	boolean	True if registered, false otherwise.
isVoiceEnabled	boolean	True if voice enabled, false otherwise.
motd	string	Fleet message of the day.

7.2.44 FleetMember

Category	Character	
Access	Character Fleets Access (ACCESS_CHARACTER_FLEETS)	
Field Name	**Type**	**Description**
fleetID	number	Unique fleet ID.
characterID	number	Character ID of fleet member.
joinTime	number	Time fleet joined (millis UTC).
joinTimeDate	string	Time fleet joined (YYYY-MM-DDTHH:MM:SS.sssZ).
role	string	Fleet member role. One of "fleet_commander", "wing_commander", "squad_commander" or "squad_member".
roleName	string	Localized role name.
shipTypeID	number	Ship type ID of ship member is flying.
solarSystemID	number	ID of solar system where member is located.
squadID	number	Squad ID for member, or -1 if member not assigned to a squad.
stationID	number	ID of station in which the member is docked, or 0 if not docked.
takesFleetWarps	boolean	True if the member takes fleet warps, false otherwise.
wingID	number	Wing ID for member, or -1 if member not assigned to a wing.

7.2.45 FleetSquad

Category	Character	
Access	Character Fleets Access (ACCESS_CHARACTER_FLEETS)	
Field Name	**Type**	**Description**
fleetID	number	Unique fleet ID.
wingID	number	Wing ID to which this squad is assigned.
squadID	number	Unique squad ID.
name	string	Squad name.

7.2.46 FleetWing

Category	Character	
Access	Character Fleets Access (ACCESS_CHARACTER_FLEETS)	
Field Name	**Type**	**Description**
fleetID	number	Unique fleet ID.
wingID	number	Unique wing ID.
name	string	Wing name.

7.2.47 Implant

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
typeID	number	Type ID of implant.

7.2.48 JumpClone

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
jumpCloneID	number	Unique jump clone ID.
locationID	number	Station or structure ID where jump clone is located.
cloneName	string	Clone name.
locationType	string	Jump clone location type. One of “station” or “structure”.

7.2.49 JumpCloneImplant

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
jumpCloneID	number	Unique jump clone ID.
typeID	number	Type ID of implant.

7.2.50 LoyaltyPoints

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
corporationID	number	Corporation ID for which loyalty points have been earned.
loyaltyPoints	number	Loyalty points earned for this corporation.

7.2.51 MailingList

Category	Character	
Access	Mailing Lists Access (ACCESS_MAILING_LISTS)	
Field Name	**Type**	**Description**
displayName	string	Mailing list name as displayed in client.
listID	number	Unique mailing list ID.

7.2.52 MailLabel

Category	Character	
Access	Mail Access (ACCESS_MAIL)	
Field Name	**Type**	**Description**
labelID	number	Unique mail label ID.
unreadCount	number	Count of unread messages for this label.
name	string	Label name.
color	string	Label color. See color.

7.2.53 MailMessageRecipient

Category	Character	
Access	Mail Access (ACCESS_MAIL)	
Field Name	**Type**	**Description**
recipientType	string	One of “alliance”, “character”, “corporation” or “mailing_list”.
recipientID	number	ID of recipient.

7.2.54 MiningLedger

Category	Character	
Access	Mining Ledger Access (ACCESS_MINING_LEDGER)	
Field Name	**Type**	**Description**
date	number	Date of mining ledger entry (millis UTC).
dateDate	string	Date of mining ledger entry (YYYY-MM-DDTHH:MM:SS.sssZ).
solarSystemID	number	ID of solar system where mining occurred.
typeID	number	ID of type of ore mined.
quantity	number	Quantity of ore mined.

7.2.55 Opportunity

Category	Character	
Access	Character Sheet Access (ACCESS_CHARACTER_SHEET)	
Field Name	**Type**	**Description**
taskID	number	Unique task ID.
completedAt	number	Date task was completed (millis UTC).
completedAtDate	string	Date task was completed (YYYY-MM-DDTHH:MM:SS.sssZ).

7.2.56 PlanetaryColony

Category	Character	
Access	Asset Access (ACCESS_ASSETS)	
Field Name	**Type**	**Description**
planetID	number	Unqiue planet ID.
solarSystemID	number	ID of solar system where planet is located.
planetType	string	Planet type. See planet_type.
ownerID	number	Character ID of colony owner.
lastUpdate	number	Last update date (millis UTC).
lastUpdateDate	string	Last update date (YYYY-MM-DDTHH:MM:SS.sssZ).
upgradeLevel	number	Colony upgrade level.
numberOfPins	number	Number of colony pins.

7.2.57 PlanetaryLink

Category	Character	
Access	Asset Access (ACCESS_ASSETS)	
Field Name	**Type**	**Description**
planetID	number	Unqiue planet ID.
sourcePinID	number	Link source pin ID.
destinationPinID	number	Link destination pin ID.
linkLevel	number	Link level.

7.2.58 PlanetaryPin

Category	Character	
Access	Asset Access (ACCESS_ASSETS)	
Field Name	**Type**	**Description**
planetID	number	Planet ID.
pinID	number	Unique pin ID.
typeID	number	Type ID of type constructed or extracted at pin.
schematicID	number	Schematic ID of factory, or 0 if not a factory pin.
lastCycleStart	number	Time when last extract cycle started (millis UTC).
lastCycleStartDate	string	Time when last extract cycle started (YYYY-MM-DDTHH:MM:SS.sssZ).
cycleTime	number	Extraction cycle time in seconds.
quantityPerCycle	number	Amount extracted per cycle.
installTime	number	Time when extraction or factory installed (millis UTC).
installTimeDate	string	Time when extraction or factory installed (YYYY-MM-DDTHH:MM:SS.sssZ).
expiryTime	number	Time when extraction expires (millis UTC).
expiryTimeDate	string	Time when extraction expires (YYYY-MM-DDTHH:MM:SS.sssZ).
productTypeID	number	ID of type extracted by extractor.
longitude	number	Longitude of pin.
latitude	number	Latitude of pin.
headRadius	number	Extractor head radius.
heads	set<*PlanetaryPinHead*>	Set of extractor heads.
contents	set<*PlanetaryPinContent*>	Current factory contents.

7.2.59 PlanetaryPinContent

Category	Character	
Access	Asset Access (ACCESS_ASSETS)	
Field Name	**Type**	**Description**
typeID	number	Type ID of content.
amount	number	Content amount.

7.2.60 PlanetaryPinHead

Category	Character	
Access	Asset Access (ACCESS_ASSETS)	
Field Name	**Type**	**Description**
headID	number	Unique head ID.
latitude	number	Head latitude.
longitude	number	Head longitude.

7.2.61 PlanetaryRoute

Category	Character	
Access	Asset Access (ACCESS_ASSETS)	
Field Name	**Type**	**Description**
planetID	number	Planet ID.
routeID	number	Unique route ID.
sourcePinID	number	Source pin ID.
destinationPinID	number	Destination pin ID.
contentTypeID	number	Type ID transmitted on this route.
quantity	number	Quantity of content transmitted on this route.
waypoints	list<number>	Ordered list of route pins IDs.

7.2.62 ResearchAgent

Category	Character	
Access	Research Access (ACCESS_RESEARCH)	
Field Name	**Type**	**Description**
agentID	number	Unique agent ID.
pointsPerDay	number	Research points generated per day.
remainderPoints	number	Remaining research points to be generated.
researchStartDate	number	Date when research started (millis UTC).
researchStartDateDate	string	Date when research started (YYYY-MM-DDTHH:MM:SS.sssZ).
skillTypeID	number	Type ID of skill used for research.

7.2.63 SkillInQueue

Category	Character	
Access	Skill Queue Access (ACCESS_SKILL_QUEUE)	
Field Name	**Type**	**Description**
endSP	number	Skill point total in the trained skill when training completes.
endTime	number	End of skill training time (millis UTC).
endTimeDate	string	End of skill training time (YYYY-MM-DDTHH:MM:SS.sssZ).
level	number	Level training to.
queuePosition	number	Position in skill training queue.
startSP	number	Skill point total in the trained skill at the previous level.
startTime	number	Start of skill training time (millis UTC).
startTimeDate	string	Start of skill training time (YYYY-MM-DDTHH:MM:SS.sssZ).
typeID	number	Type ID of skill being trained.
trainingStartSP	number	Skill point total in the trained skill when training started.

7.2.64 UpcomingCalendarEvent

Category	Character	
Access	Upcoming Calendar Event Access (ACCESS_UPCOMING_CALENDAR_EVENTS)	
Field Name	**Type**	**Description**
duration	number	Event duration in minutes.
eventDate	number	Date of event (millis UTC).
eventDateDate	string	Date of event (YYYY-MM-DDTHH:MM:SS.sssZ).
eventID	number	Unique event ID.
eventText	string	Description of event.
eventTitle	string	Event title.
ownerID	number	Event owner ID.
ownerName	string	Event owner name.
response	string	Response to event invitation.
importance	number	Event importance level.
ownerType	string	Owner type. One of “eve_server”, “corporation”, “faction”, “character” or “alliance”.

7.2.65 ContainerLog

Category	Corporation	
Access	Container Log Access (ACCESS_CONTAINER_LOG)	
Field Name	**Type**	**Description**
logTime	number	Log time (millis UTC).
logTimeDate	string	Log time (YYYY-MM-DDTHH:MM:SS.sssZ).
action	string	Container action. See action.
characterID	number	Character ID of character which accessed the container.
locationFlag	string	Container location flag. See location_flag.
containerID	number	Unique container ID.
containerTypeID	number	Type ID of container.
locationID	number	Container location ID.
newConfiguration	number	New configuration mask.
oldConfiguration	number	Old configuration mask.
passwordType	string	Password type. either “config” or “general”.
quantity	number	Quantity of container items acted on.
typeID	number	Type ID of item acted on.

7.2.66 CorporationMedal

Category	Corporation	
Access	Corporation Medals Access (ACCESS_CORPORATION_MEDALS)	
Field Name	**Type**	**Description**
medalID	number	Unique medal ID.
description	string	Medal description.
title	string	Medal title.
created	number	Medal creation date (millis UTC).
createdDate	string	Medal creation date (YYYY-MM-DDTHH:MM:SS.sssZ).
creatorID	number	Character ID of medal creator.

7.2.67 CorporationMemberMedal

Category	Corporation	
Access	Member Medals Access (ACCESS_MEMBER_MEDALS)	
Field Name	**Type**	**Description**
medalID	number	Unique medal ID.
characterID	number	ID of character receiving medal.
issued	number	Date medal issued (millis UTC).
issuedDate	string	Date medal issued (YYYY-MM-DDTHH:MM:SS.sssZ).
issuerID	number	ID of character who issued medal.
reason	string	Reason medal was issued.
status	string	Medal status. One of “private” or “public”.

7.2.68 CorporationSheet

Category	Corporation	
Access	Corporation Sheet Access (ACCESS_CORPORATION_SHEET)	
Field Name	**Type**	**Description**
allianceID	number	ID of alliance of which corporation is a member, or 0 if not in an alliance.
ceoID	number	Character ID of corporation CEO.
corporationID	number	Unique corporation ID.
corporationName	string	Corporation name.
description	string	Corporation description.
memberCount	number	Current corporation member count.
shares	number	Current corporation share count.
stationID	number	Corporation home station ID.
taxRate	number	Corporation tax rate (between 0 and 1).
ticker	string	Corporation in-game ticker.
url	string	Corporation home page URL.
dateFounded	number	Date of corporation founding (millis UTC).
dateFoundedDate	string	Date of corporation founding (YYYY-MM-DDTHH:MM:SS.sssZ).
creatorID	number	Characger ID of corporation founder.
factionID	number	ID of faction of which corporation is a member, or 0 if not in a faction.
px64x64	string	64x64 corporation logo URL.
px128x128	string	128x128 corporation logo URL.
px256x256	string	256x256 corporation logo URL.

7.2.69 CorporationTitle

Category	Corporation	
Access	Corporation Titles Access (ACCESS_CORPORATION_TITLES)	
Field Name	**Type**	**Description**
titleID	number	Unique title ID.
titleName	string	Title name.

7.2.70 CorporationTitleRole

Category	Corporation	
Access	Corporation Titles Access (ACCESS_CORPORATION_TITLES)	
Field Name	**Type**	**Description**
titleID	number	Unique title ID.
roleName	string	Title role name.
grantable	boolean	True if role is grantable, false otherwise.
atHQ	boolean	True if role is at HQ, false otherwise.
atBase	boolean	True if role is at a base, false otherwise.
atOther	boolean	True if role is at other, false otherwise.

7.2.71 CustomsOffice

Category	Corporation	
Access	Asset Access (ACCESS_ASSETS)	
Field Name	**Type**	**Description**
officeID	number	Unique customs office ID.
solarSystemID	number	ID of solar system where customs office located.
reinforceExitStart	number	Starting hour for the 2-hour reinforcement exit window after an attack.
reinforceExitEnd	number	Ending hour for the 2-hour reinforcement exit window after an attack.
allowAlliance	boolean	True if alliance access allowed, false otherwise.
allowStandings	boolean	True if access allowed based on standings, false otherwise.
standingLevel	string	If "allowStandings" is true, then access is only allowed by entities with this level of standing or better. One of "bad", "excellent", "good", "neutral" or "terrible".
taxRateAlliance	number	Tax rate for alliance members.
taxRateCorp	number	Tax rate for corporation members.
taxRateStandingExcellent	number	Tax rate for characters with excellent standing.
taxRateStandingGood	number	Tax rate for characters with good standing.
taxRateStandingNeutral	number	Tax rate for characters with neutral standing.
taxRateStandingBad	number	Tax rate for characters with bad standing.
taxRateStandingTerrible	number	Tax rate for characters with terrible standing.

7.2.72 Division

Category	Corporation	
Access	Corporation Sheet Access (ACCESS_CORPORATION_SHEET)	
Field Name	**Type**	**Description**
wallet	boolean	True if this is a wallet division, false otherwise.
division	number	Unique division ID (1-7).
name	string	Division name.

7.2.73 Facility

Category	Corporation	
Access	Industry Jobs Access (ACCESS_INDUSTRY_JOBS)	
Field Name	**Type**	**Description**
facilityID	number	Unique facility ID.
typeID	number	Type ID of facility structure.
solarSystemID	number	ID of solar system where facility located.

7.2.74 Fuel

Category	Corporation	
Access	Starbase List Access (ACCESS_STARBASE_LIST)	
Field Name	**Type**	**Description**
starbaseID	number	Unique starbase ID.
typeID	number	Fuel type ID.
quantity	number	Fuel quantity.

7.2.75 Member

Category	Corporation	
Access	Member Security Access (ACCESS_MEMBER_SECURITY)	
Field Name	**Type**	**Description**
characterID	number	Character ID of member.

7.2.76 MemberLimit

Category	Corporation	
Access	Member Tracking Access (ACCESS_MEMBER_TRACKING)	
Field Name	**Type**	**Description**
memberLimit	number	Current corporation member limit.

7.2.77 MemberRole

Category	Corporation	
Access	Member Security Access (ACCESS_MEMBER_SECURITY)	
Field Name	**Type**	**Description**
characterID	number	ID of character to which role is assigned.
roleName	string	Name of assigned role.
grantable	boolean	True if this is a grantable role, false otherwise.
atHQ	boolean	True if this role is at HQ, false otherwise.
atBase	boolean	True if this role is at a base, false otherwise.
atOther	boolean	True if this role is at other, false otherwise.

7.2.78 MemberRoleHistory

Category	Corporation	
Access	Member Security Log Access (ACCESS_MEMBER_SECURITY_LOG)	
Field Name	**Type**	**Description**
characterID	number	ID of character for which role changed.
changedAt	number	Date of role change (millis UTC).
changedAtDate	string	Date of role change (YYYY-MM-DDTHH:MM:SS.sssZ).
issuerID	number	Character ID who changed role.
roleType	string	Role type.
roleName	string	Role name.
old	boolean	If true, then this update is describing the role before the change. Otherwise, this update describes the role after the change.

7.2.79 MemberTitle

Category	Corporation	
Access	Member Security Access (ACCESS_MEMBER_SECURITY)	
Field Name	**Type**	**Description**
characterID	number	ID of character to which title is assigned.
titleID	number	ID of assigned title.

7.2.80 MemberTracking

Category	Corporation	
Access	Member Tracking Access (ACCESS_MEMBER_TRACKING)	
Field Name	**Type**	**Description**
characterID	number	Character ID of member.
baseID	number	ID of character's base.
locationID	number	ID of character's current location.
logoffDateTime	number	Last logoff time (millis UTC).
logoffDateTimeDate	string	Last logoff time (YYYY-MM-DDTHH:MM:SS.sssZ).
logonDateTime	number	Last logon time (millis UTC).
logonDateTimeDate	string	Last logon time (YYYY-MM-DDTHH:MM:SS.sssZ).
shipTypeID	number	Character's ship type ID.
startDateTime	number	Character membership start time (millis UTC).
startDateTimeDate	string	Character membership start time (YYYY-MM-DDTHH:MM:SS.sssZ).

7.2.81 MiningExtraction

Category	Corporation	
Access	Mining Ledger Access (ACCESS_MINING_LEDGER)	
Field Name	**Type**	**Description**
moonID	number	ID of moon where chunk will be extracted.
structureID	number	ID of structure performing extraction.
extractionStartTime	number	Time when extraction started (millis UTC).
extractionStartTimeDate	string	Time when extraction started (YYYY-MM-DDTHH:MM:SS.sssZ).
chunkArrivalTime	number	Time when chunk will arrive (millis UTC).
chunkArrivalTimeDate	string	Time when chunk will arrive (YYYY-MM-DDTHH:MM:SS.sssZ).
naturalDecayTime	number	Time when chunk will decay (millis UTC).
natrualDecayTimeDate	string	Time when chunk will decay (YYYY-MM-DDTHH:MM:SS.sssZ).

7.2.82 MiningObservation

Category	Corporation	
Access	Mining Ledger Access (ACCESS_MINING_LEDGER)	
Field Name	**Type**	**Description**
observerID	number	ID of the entity that observed the mining.
characterID	number	ID of character that performed the mining.
typeID	number	Type ID of item mined.
recordedCorporationID	number	Corporation ID of the mining character at the time when the observation was recorded.
quantity	number	Volume mined.
lastUpdated	number	Observation time (millis UTC).
lastUpdatedDate	string	Observation time (YYYY-MM-DDTHH:MM:SS.sssZ).

7.2.83 MiningObserver

Category	Corporation	
Access	Mining Ledger Access (ACCESS_MINING_LEDGER)	
Field Name	**Type**	**Description**
observerID	number	ID of the observing entity.
observerType	string	Observer type. Currently always "structure".
lastUpdated	number	Observer update time (millis UTC).
lastUpdateDate	string	Observer update time (YYYY-MM-DDTHH:MM:SS.sssZ).

7.2.84 Shareholder

Category	Corporation	
Access	Shareholder Access (ACCESS_SHAREHOLDERS)	
Field Name	**Type**	**Description**
shareholderID	number	ID of shareholder entity.
shareholderType	string	Shareholder type. Currently either "character" or "corporation".
shares	number	Number of shares held by shareholder.

7.2.85 Starbase

Category	Corporation	
Access	Starbase List Access (ACCESS_STARBASE_LIST)	
Field Name	**Type**	**Description**
starbaseID	number	Unique starbase ID.
typeID	number	Starbase type ID.
systemID	number	ID of solar system where starbase is located.
moonID	number	ID of moon starbase is orbiting.
state	string	Starbase state. See state.
unanchorAt	number	Time when starbase unanchored (millis UTC).
unanchorAtDate	string	Time when starbase unanchored (YYYY-MM-DDTHH:MM:SS.sssZ).
reinforcedUntil	number	Time up to which starbase will be reinforced (millis UTC).
reinforcedUntilDate	string	Time up to which starbase will be reinforced (YYYY-MM-DDTHH:MM:SS.sssZ).
onlinedSince	number	Time at which starbase was onlined (millis UTC).
onlinedSinceDate	string	Time at which starbase was onlined (YYYY-MM-DDTHH:MM:SS.sssZ).
fuelBayView	string	Roles allowed to view the starbase fuel bay. See fuel_bay_view.
fuelBayTake	string	Roles allowed to take fuel from the starbase fuel bay. See fuel_bay_take.
anchor	string	Roles allowed to anchor the starbase. See anchor.
unanchor	string	Roles allowed to unanchor the starbase. See unanchor.
online	string	Roles allowed to online the starbase. See online.
offline	string	Roles allowed to offline the starbase. See offline.
allowCorporationMembers	boolean	True if corporation members allowed access, false otherwise.
allowAllianceMembers	boolean	True if alliance members allowed access, false otherwise.
useAllianceStandings	boolean	True if alliance standings should be used for threshold checks, false otherwise.
attackStandingThreshold	number	Threshold for checking attack on standing.
attackSecurityStatusThreshold	number	Threshold for checking att on security status.
attackIfOtherSecurityStatusDropping	boolean	True if starbase will attack accessing entities with dropping security status, false otherwise.
attackIfWar	boolean	True if starbase will attack accessing entities at war with owner, false otherwise.

7.2.86 Structure

Category	Corporation	
Access	Structure List Access (ACCESS_STRUCTURES)	
Field Name	**Type**	**Description**
structureID	number	Unique structure ID.
corporationID	number	ID of corporation which owns structure.
fuelExpires	number	Fuel expire date (millis UTC).
fuelExpiresDate	string	Fuel expire date (YYYY-MM-DDTHH:MM:SS.sssZ).
nextReinforceApply	number	Time when new reinforce hour and weekday will take effect (millis UTC).
nextReinforceApplyDate	string	Time when new reinforce hour and weekday will take effect (YYYY-MM-DDTHH:MM:SS.sssZ).
nextReinforceHour	number	New requested reinforce hour.
nextReinforceWeekday	number	New requested reinforce weekday.
profileID	number	Access Control List (ACL) profile ID for this structure.
reinforceHour	number	Hour of the day that marks the mid-point of the four hour window in which the structure will randomly exit reinforcement.
reinforceWeekday	number	Day of the week when structure exits final reinforcement. One of 0 (Monday) through 6 (Sunday).
state	string	Structure state. See state.
stateTimerEnd	number	Time at which structure will enter its next state (millis UTC).
stateTimerEndDate	string	Time at which structure will enter its next state (YYYY-MM-DDTHH:MM:SS.sssZ).
stateTimerStart	number	Time at which structure entered its current state (millis UTC).
stateTimerStartDate	string	Time at which structure entered its current state (YYYY-MM-DDTHH:MM:SS.sssZ).
systemID	number	ID of solar system where structure is located.
typeID	number	Structure type ID.
unanchorsAt	number	Time at which structure will unanchor (millis UTC).
unanchorsAtDate	string	Time at which structure will unanchor (YYYY-MM-DDTHH:MM:SS.sssZ).

7.2.87 StructureService

Category	Corporation	
Access	Structure List Access (ACCESS_STRUCTURES)	
Field Name	**Type**	**Description**
structureID	number	Structure ID to which this service is attached.
name	string	Service name.
state	string	Service state. One of “online”, “offline” or “cleanup”.

CHAPTER 8

Model Synchronization and Access

EveKit data is updated through a process called model synchronization. During model synchronization, data is retrieved from ESI endpoints and compared to the current live version of various data models. If new data is encountered, or previous data has been removed, then EveKit's model objects are updated as described in the previous section. EveKit attempts to look for new data at least as frequently as the ESI cache timers allow, and as often as load on the EveKit servers allow. At time of writing, the shortest interval between EveKit updates is 7 minutes. The fastest ESI cache timer is about five seconds (e.g. character location).

8.1 Endpoint to Model Data Mapping

The following table shows the mapping from ESI endpoint(s) to EveKit model object(s).

ESI endpoint	EveKit Models Updated	Notes
• /alliances/{alliance_id}/contacts/ • /alliances/{alliance_id}/contacts/labels/ • /characters/{character_id}/contacts/labels/ • /characters/{character_id}/contacts/ • /corporations/{corporation_id}/contacts/labels/ • /corporations/{corporation_id}/contacts/	• *Contact* • *ContactLabel*	Contacts and contact labels are updated simultaneously during synchronization.

Continued on next page

Table 1 – continued from previous page

• /characters/{character_id}/assets/ • /characters/{character_id}/assets/locations/ • /characters/{character_id}/assets/names/ • /corporations/{corporation_id}/assets/ • /corporations/{corporation_id}/assets/locations/ • /corporations/{corporation_id}/assets/names/	• *Asset* • *Location*	Assets, their locations and their names are updated simultaneously during synchronization. Note that *Location* stores both location and name.
• /characters/{character_id}/blueprints/ • /corporations/{corporation_id}/blueprints/	• *Blueprint*	
• /characters/{character_id}/bookmarks/ • /characters/{character_id}/bookmarks/folders/ • /corporations/{corporation_id}/bookmarks/ • /corporations/{corporation_id}/bookmarks/folders/	• *Bookmark*	EveKit does not store a separate bookmark folder object. Instead, bookmark and folder information are combined in the *Bookmark* object. A consequence is that empty folder names are not stored. This may change in the future.
• /characters/{character_id}/calendar/ • /characters/{character_id}/calendar/{event_id}/ • /characters/{character_id}/calendar/{event_id}/attendees/	• *UpcomingCalendarEvent* • *CalendarEventAttendee*	Calendar events and their attendees are updated simultaneously during synchronization.
• /characters/{character_id}/contracts/ • /characters/{character_id}/contracts/{contract_id}/bids/ • /characters/{character_id}/contracts/{contract_id}/items/ • /corporations/{corporation_id}/contracts/ • /corporations/{corporation_id}/contracts/{contract_id}/bids/ • /corporations/{corporation_id}/contracts/{contract_id}/items/	• *Contract* • *ContractItem* • *ContractBid*	Contracts, contract items and contract bids (for auctions) are updated simultaneously during synchronization.

Continued on next page

Table 1 – continued from previous page

• /characters/{character_id}/fw/stats/ • /corporations/{corporation_id}/fw/stats/	• *FacWarStats*	
• /characters/{character_id}/fittings/	• *Fitting* • *FittingItem*	
• /characters/{character_id}/fleet/ • /fleets/{fleet_id}/ • /fleets/{fleet_id}/members/ • /fleets/{fleet_id}/wings/	• *CharacterFleet* • *FleetInfo* • *FleetMember* • *FleetWing* • *FleetSquad*	Character fleet information including fleet composition is updated simultaneously during synchronization.
• /characters/{character_id}/industry/jobs/ • /corporations/{corporation_id}/industry/jobs/	• *IndustryJob*	
• /characters/{character_id}/killmails/recent/ • /corporations/{corporation_id}/killmails/recent/ • /killmails/{killmail_id}/{killmail_hash}/	• *Kill* • *KillItem* • *KillVictim* • *KillAttacker*	Kill mail processing is sharded by killmail ID due to size. As a result, several synchronization rounds are necessary to process all recent killmail. At time of writing, 10 rounds are necessary to completely process all recent kill mail. During a round, the sharded kill list and detailed kill information are updated simultaneously.
• /characters/{character_id}/location/	• *CharacterLocation*	
• /characters/{character_id}/loyalty/points/	• *LoyaltyPoints*	

Continued on next page

Table 1 – continued from previous page

• /characters/{character_id}/mail/ • /characters/{character_id}/mail/{mail_id}/ • /characters/{character_id}/mail/lists/ • /characters/{character_id}/mail/labels/	• *CharacterMailMessage* • *MailingList* • *MailLabel*	Mail processing is sharded by mail ID due to size. As a result, several synchronization rounds are necessary to process all mail. At time of writing, 20 rounds are necessary to completely process all mail. During a round, the sharded mail list and detailed mail information are updated simultaneously. Mailing list and mail label processing is not sharded and will be updated in a single synchronization round.
• /characters/{character_id}/orders/ • /characters/{character_id}/orders/history/ • /corporations/{corporation_id}/orders/ • /corporations/{corporation_id}/orders/history/	• *MarketOrder*	Market orders and market order history are processed simultaneously during synchronization.
• /characters/{character_id}/medals/	• *CharacterMedal* • *CharacterMedalGraphic*	
• /characters/{character_id}/mining/	• *MiningLedger*	
• /characters/{character_id}/notifications/ • /characters/{character_id}/notifications/contacts/	• *CharacterContactNotification* • *CharacterNotification*	Notifications and contact notifications are processed simultaneously during synchronization.
• /characters/{character_id}/online/	• *CharacterOnline*	
• /characters/{character_id}/opportunities/	• *Opportunity*	
• /characters/{character_id}/planets/ • /characters/{character_id}/planets/{planet_id}/	• *PlanetaryColony* • *PlanetaryRoute* • *PlanetaryLink* • *PlanetaryPin*	The planet list and detailed planet information are processed simultaneously during synchronization.

Continued on next page

Table 1 – continued from previous page

• /characters/{character_id}/agents_research/	• *ResearchAgent*	
• /characters/{character_id}/roles/	• *CharacterRole*	
• /characters/{character_id}/clones/	• *CharacterSheetClone* • *JumpClone* • *JumpCloneImplant*	
• /characters/{character_id}/implants/	• *Implant*	
• /characters/{character_id}/fatigue/	• *CharacterSheetJump*	
• /characters/{character_id}/	• *CharacterSheet*	
• /characters/{character_id}/ship/	• *CharacterShip*	
• /characters/{character_id}/skillqueue/	• *SkillInQueue*	
• /characters/{character_id}/skills/ • /characters/{character_id}/attributes/	• *CharacterSheetAttributes* • *CharacterSheetSkillPoints* • *CharacterSkill*	A character's skill list and attributes are processed simultaneously during synchronization.
• /characters/{character_id}/standings/ • /corporations/{corporation_id}/standings/	• *Standing*	
• /corporations/{corporation_id}/titles/	• *CharacterTitle*	

Continued on next page

Table 1 – continued from previous page

• /characters/{character_id}/wallet/ • /corporations/{corporation_id}/wallets/	• *AccountBalance*	
• /characters/{character_id}/wallet/journal/ • /corporations/{corporation_id}/wallets/{division}/journal/	• *WalletJournal*	
• /characters/{character_id}/wallet/transactions/ • /corporations/{corporation_id}/wallets/{division}/transactions/	• *WalletTransaction*	
• /corporations/{corporation_id}/containers/logs/	• *ContainerLog*	
• /corporations/{corporation_id}/customs_offices/	• *CustomsOffice*	
• /corporations/{corporation_id}/divisions/	• *Division*	
• /corporations/{corporation_id}/facilities/	• *Facility*	
• /corporations/{corporation_id}/medals/ • /corporations/{corporation_id}/medals/issued/	• *CorporationMedal* • *CorporationMemberMedal*	Corporation medals and issued medals are processed simultaneously during synchronization.
• /corporations/{corporation_id}/members/ • /corporations/{corporation_id}/roles/ • /corporations/{corporation_id}/roles/history/	• *MemberRole* • *MemberRoleHistory* • *Member*	Corporation member list, roles and history of role changes are processed simultaneously during synchronization.

Continued on next page

Table 1 – continued from previous page

• /corporations/{corporation_id}/membertracking/	• *MemberLimit* • *MemberTracking*	Corporation member limit and member tracking are processed simultaneously during synchronization.
• /corporation/{corporation_id}/mining/extractions/ • /corporation/{corporation_id}/mining/observers/ • /corporation/{corporation_id}/mining/observers/{observer_id}/	• *MiningExtraction* • *MiningObserver* • *MiningObservation*	Mining extractions, observer list, and observations are processed simultaneously during synchronization.
• /corporations/{corporation_id}/shareholders/	• *Shareholder*	
• /corporations/{corporation_id}/ • /corporations/{corporation_id}/icons/	• *CorporationSheet*	Corporation sheet and icons are processed simultaneously during synchronization.
• /corporations/{corporation_id}/starbases/ • /corporations/{corporation_id}/starbases/{starbase_id}/	• *Starbase* • *Fuel*	Starbase list and starbase details are processed simultaneously during synchronization.
• /corporations/{corporation_id}/structures/	• *Structure* • *StructureService*	
• /corporations/{corporation_id}/titles/ • /corporations/{corporation_id}/members/titles/	• *CorporationTitle* • *CorporationTitleRole* • *MemberTitle*	Corporation titles, roles and titles assigned to members are processed simultaneously during synchronization.

8.2 Access Permission to Model Data Mapping

The following table gives the mapping from EveKit Access Key Permissions to the model data objects which can be retrieved with those permissions.

Permission	Mask	EveKit Models

Continued on next page

Table 2 – continued from previous page

Account Status Access	ACCESS_ACCOUNT_STATUS	• *CharacterOnline*
Account Balance Access	ACCESS_ACCOUNT_BALANCE	• *AccountBalance*
Asset Access	ACCESS_ASSETS	• *Asset*
Character Sheet Access	ACCESS_CHARACTER_SHEET	• *CharacterRole* • *CharacterSheet* • *CharacterSheetAttributes* • *CharacterSheetClone* • *CharacterSheetJump* • *CharacterSheetSkillPoints* • *CharacterSkill* • *CharacterTitle* • *Implant* • *JumpClone* • *JumpCloneImplant* • *LoyaltyPoints* • *Opportunity*
Corporation Sheet	ACCESS_CORPORATION_SHEET	• *CorporationSheet* • *Division*
Contact List Access	ACCESS_CONTACT_LIST	• *Contact* • *ContactLabel*
Blueprint Access	ACCESS_BLUEPRINTS	• *Blueprint*
Bookmark Access	ACCESS_BOOKMARKS	• *Bookmark*
Contract Access	ACCESS_CONTRACTS	• *Contract* • *ContractBid* • *ContractItem*
Faction War Stats Access	ACCESS_FAC_WAR_STATS	• *FacWarStats*
Industry Jobs Access	ACCESS_INDUSTRY_JOBS	• *IndustryJob* • *Facility*

Continued on next page

Table 2 – continued from previous page

Kill Log Access	ACCESS_KILL_LOG	• *Kill* • *KillAttacker* • *KillItem* • *KillVictim*
Locations Access	ACCESS_LOCATIONS	• *CharacterLocation* • *CharacterShip* • *Location*
Market Orders Access	ACCESS_MARKET_ORDERS	• *MarketOrder*
Mining Ledger Access	ACCESS_MINING_LEDGER	• *MiningLedger* • *MiningExtraction* • *MiningObservation* • *MiningObserver*
Standings Access	ACCESS_STANDINGS	• *Standing*
Wallet Journal Access	ACCESS_WALLET_JOURNAL	• *WalletJournal*
Wallet Transactions Access	ACCESS_WALLET_TRANSACTIONS	• *WalletTransaction*
Meta-Data Modification	ALLOW_METADATA_CHANGES	N/A - This mask allows write access to the meta-data map for any model object for which the access key has read access.
Calendar Event Attendee Access	ACCESS_CALENDAR_EVENT_ATTENDEES	• *CalendarEventAttendee*
Access Chat Channels	ACCESS_CHAT_CHANNELS	• *ChatChannel (deprecated)* • *ChatChannelMember (deprecated)*
Access Character Fleets	ACCESS_CHARACTER_FLEETS	• *CharacterFleet* • *FleetInfo* • *FleetMember* • *FleetSquad* • *FleetWing*
Access Contact Notifications	ACCESS_CONTACT_NOTIFICATIONS	• *CharacterContactNotification*

Continued on next page

Table 2 – continued from previous page

Access Ship Fittings	ACCESS_FITTINGS	• *Fitting* • *FittingItem*
Access Mail	ACCESS_MAIL	• *CharacterMailMessage* • *MailLabel*
Access Mailing Lists	ACCESS_MAILING_LISTS	• *MailingList*
Access Medals	ACCESS_MEDALS	• *CharacterMedal* • *CharacterMedalGraphic*
Access Notifications	ACCESS_NOTIFICATIONS	• *CharacterNotification*
Access Research	ACCESS_RESEARCH	• *ResearchAgent*
Access Skill Queue	ACCESS_SKILL_QUEUE	• *SkillInQueue*
Upcoming Calendar Event Access	ACCESS_UPCOMING_CALENDAR_EVENTS	• *UpcomingCalendarEvent*
Container Log Access	ACCESS_CONTAINER_LOG	• *ContainerLog*
Corporation Medals Access	ACCESS_CORPORATION_MEDALS	• *CorporationMedal*
Member Medals Access	ACCESS_MEMBER_MEDALS	• *CorporationMemberMedal*
Member Security Access	ACCESS_MEMBER_SECURITY	• *Member* • *MemberRole* • *MemberTitle*
Member Security Log Access	ACCESS_MEMBER_SECURITY_LOG	• *MemberRoleHistory*
Member Tracking Access	ACCESS_MEMBER_TRACKING	• *MemberLimit* • *MemberTracking*
Shareholder Access	ACCESS_SHAREHOLDERS	• *Shareholder*

Continued on next page

Table 2 – continued from previous page

Starbase List Access	ACCESS_STARBASE_LIST	• *Fuel* • *Starbase*
Structure List Access	ACCESS_STRUCTURES	• *Structure* • *StructureService*
Corporation Titles Access	ACCESS_CORPORATION_TITLES	• *CorporationTitle* • *CorporationTitleRole*

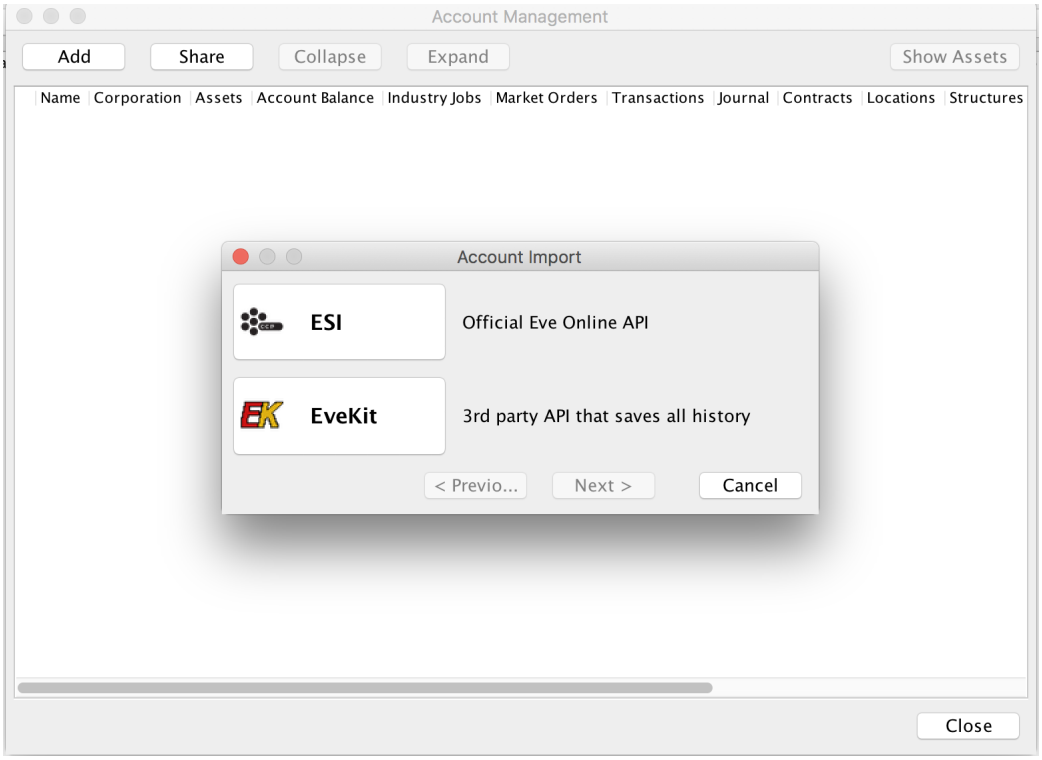
CHAPTER 9

Using EveKit with jEveAssets

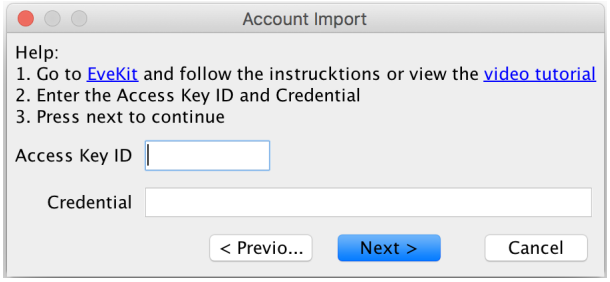
jEveAssets is an out-of-game management tool for EVE Online. This tool is written in Java and is easy to install on most popular operating systems. jEveAssets operates on player and corporation data retrieved from the ESI. However, jEveAssets can also retrieve data from EveKit. Using EveKit as the source in place of the ESI ensures that a complete history of your date is always available to jEveAssets. Follow the instructions below to add you EveKit accounts to jEveAssets. This guide assumes you have successfully installed jEveAssets.

To add an EveKit account to jEveAssets, you'll first need to record the credentials of the approriate EveKit data access key. You can find these credentials in the *Model Access Keys* section of the appropriate EveKit account. If you haven't created a data access key yet, please go ahead and create one, adding the access masks for the data you would like to make visible to jEveAssets. You'll need to note the key ID and hash string for the access key you would like to import.

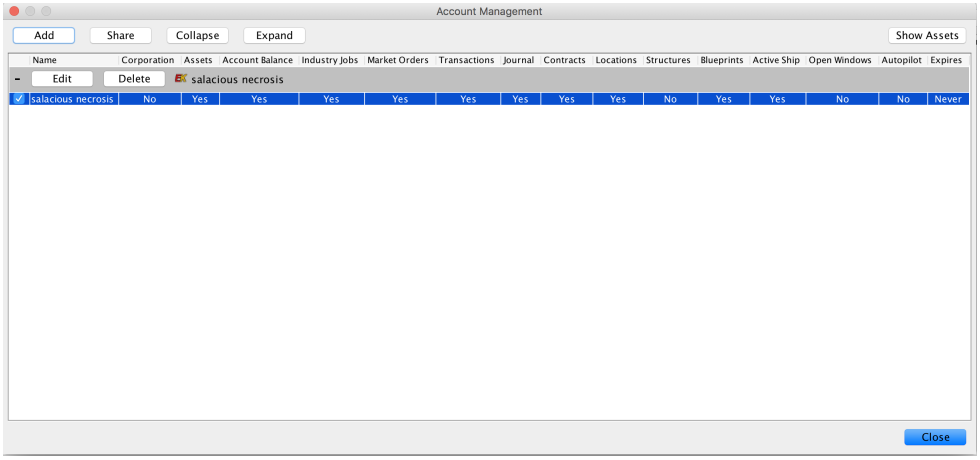
Once you have recorded your access key credentials, start jEveAssets and select Options -> Accounts. This will cause the following dialog to appear:



Click the "EveKit" button to bring up the EveKit credentials dialog:

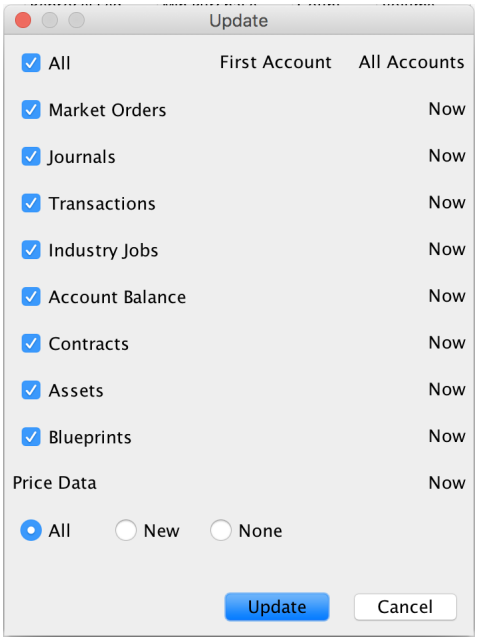


Now enter your data access key ID and hash string into the appropriate fields and click "Next". jEveAssets will spend a moment validating your key, after which you can click "OK" to dismiss the dialog. If all goes well, the "Account Management" display should look similar to the following:

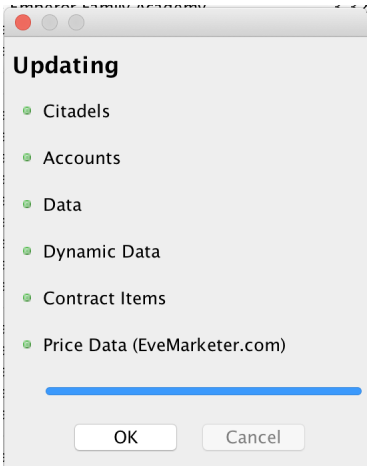


The columns in this view show data categories you have permissioned jEveAssets to retrieve. You can add additional accounts if you wish using the "Add" button. Click "Close" when you are finished adding accounts.

EveKit data will automatically be imported then next time you use the "Update" dialog. When you select Update -> Update..., you should see a dialog like the following:

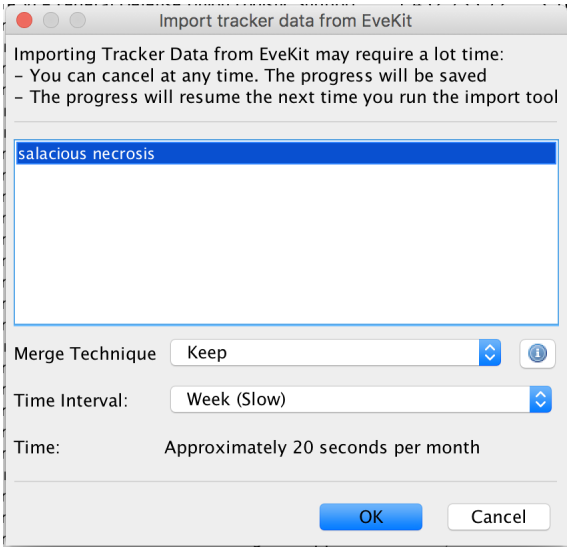


Go ahead and click "Update" to initiate data retrieval. If everything is working properly, you should see a dialog like the following:



If you close this dialog and pull up some of the jEveAssets tools, you should see the latest view of whatever data you have chosen to share with jEveAssets (e.g. asset list, wallet transactions and journal, etc).

The "Update" dialog will always retrieve the latest data from EveKit. To fully take advantage of EveKit, however, you may want to retrieve historical data as well. jEveAssets provides support for historical retrieval using the "EveKit tracker import" function. To use this feature, select Update -> EveKit tracker import. This will bring up a dialog like the following:



You can choose to import daily, weekly or monthly historical snapshots merged according to your chosen "Merge Technique". If you don't have much history, or your account is not very active, then daily snapshots should be fine. Otherwise, you should consider choosing a more coarse snapshot interval.

jEveAssets is under active development, with most new features including EveKit support. You can view the jEveAssets documentation for further information.

CHAPTER 10

Market Data

As described in *Market API*, the EveKit Market Data API consists of two services:

1. An *online* service for retrieving book and market history snapshots of individual types and regions; and,
2. An *archive* which stores order books and market history in bulk format.

In this section, we describe the services provided by the archive.

10.1 Overview

EveKit archives three forms of market data:

1. *Market History* records price and volume data for each type in each region on each day. This data is retrieved from the `/markets/{region_id}/history/` ESI endpoint.
2. *Order Books* record snapshots of the market order book for each type in each region at five minute intervals during each day. This data is retrieved from the `/markets/{region_id}/orders/` ESI endpoint. Five minute snapshots are recorded because this is the minimum resolution (e.g. cache timer) allowed by the ESI endpoint for this data.
3. *Inferred Trades* are the estimated set of trades (e.g. volume, price and direction - buy or sell) which occurred for a given type in a given region on a given day. The ESI does not provide this data. Instead, EveKIt attempts to infer the set of trades based on changes to the order book during the day.

All market data is archived to Google storage and is available at the following prefix:

```
https://storage.googleapis.com/evekit_md/YYYY/MM/DD
```

where *YYYY*, *MM* and *DD*, are the requested year, month and day, respectively.

The following table describes the set of files stored for each day:

File Name	Typical Size	Description
market_YYYYMMDD.tgz	Less than 5 MB.	All market history for all types and regions for the given day.
interval_YYYYMMDD_200.gz	About 200 MB.	All order book data for all types and regions for the given day. The value, *5*, in the name of the file indicates that these are 5 minute snapshots.
trades_allregions_YYYYMMDD.gz	Less than 5 MB.	All inferred trades for all types and regions for the given day.
market_YYYYMMDD.bulk	Less than 5 MB.	Market history for all types and regions in *bulk* format, designed to be partially retrieved via an HTTP "range" request.
interval_YYYYMMDD_500.bulk	About 500 MB.	Order book data for all types and regions in *bulk* format, designed to eb partially retrieved via an HTTP "range" request. The value, *5*, in the name of the file indicates that these are 5 minute snapshots.
market_YYYYMMDD.index.gz	Less than 100 KB.	Index file which provides offsets into the *bulk* market history file based on EVE market type ID.
interval_YYYYMMDD_5.index.gz	Less than 200 KB.	Index file which provides offsets into the *bulk* order book file based on EVE market type ID. The value, *5*, in the name of the file indicates that these are 5 mnute snapshots.

Files (and segments of files) are gzip compressed to save space. Files are generated on a schedule as follows (*t* is the date for which market data is recorded):

File Type	Availability	Notes
Market History	*t+2* 0800 UTC	Market history is not available via ESI until approximately *t+1* 1200 UTC at which point it can be collected and processed by EveKit.
Order Book	*t+1* 0800 UTC	Assembly starts at approximately *t+1* 0200 UTC
Inferred Trades	*t+1* 2000 UTC	Trades can not be inferred until order books have been processed.

We describe the format of each file type below.

10.2 Market History

Market history data records the order count, high price, low price, average price and volume for a given type in a given region. The market history archive consists of collection of files in CSV format, one file per market type ID, with one line per region as follows:

Field	Description
TYPE	EVE market type ID.
REGION	EVE market region ID.
ORDER COUNT	Number of market orders executed for this type in this region on this day.
LOW PRICE	Low trade price for this type in this region on this day.
HIGH PRICE	High trade price for this type in this region on this day.
AVERAGE PRICE	Average trade price for this type in this region on this day.
VOLUME	Daily volume for this type in this region.
DATE	Market history date in milliseconds UTC (since the epoch).

Files of the form *market_YYYYMMDD.tgz* are a compressed archive containing all history files for all types for a given day. You should retrieve this file if you wish to operate on a large number of market types for a given day. A sample listing for a recent file looks as follows:

```
$ tar tzf market_20180929.tgz | head -n 10
market_18_20180929.history.gz
market_19_20180929.history.gz
market_20_20180929.history.gz
market_21_20180929.history.gz
market_22_20180929.history.gz
market_34_20180929.history.gz
market_35_20180929.history.gz
market_36_20180929.history.gz
market_37_20180929.history.gz
market_38_20180929.history.gz
```

At time of writing, there are about 8600 files (e.g. EVE market types) in a given archive. Each file in the archive is a compressed CSV file in the format shown in the table above. The first file in this example appears as follows:

```
$ gzcat market_18_20180929.history.gz | head -n 10
18,10000028,2,56.01,56.01,56.01,9269,1538179200000
18,10000030,40,30.00,51.01,51.01,979837,1538179200000
18,10000016,92,59.36,61.02,60.80,2769268,1538179200000
18,10000064,79,52.30,52.30,52.30,4939877,1538179200000
18,10000002,135,58.29,58.31,58.29,6972853,1538179200000
18,10000068,94,53.50,53.50,53.50,16821756,1538179200000
18,10000069,1,58.00,58.00,58.00,111,1538179200000
18,10000048,4,52.01,52.01,52.01,434038,1538179200000
18,10000042,62,55.01,55.01,55.01,7323977,1538179200000
18,10000043,3,56.81,56.81,56.81,133,1538179200000
```

In most cases, market history files are small enough to simply download and extract. However, some applications (e.g. web sites) may have insufficient local storage to make this possible. For these cases, the *bulk* files can be used to extract just the data of interest. Extracting data from a bulk file involves two steps:

1. Read an index file and find the offset of the desired market type; then,
2. Use a "range" HTTP request to read the appropriate data from the bulk file.

Index files are formatted as a compressed list of pairs giving each file name and an offset into the corresponding bulk file where data for the given file is stored. Using the example from above, we can pull the first few lines from the index file as follows:

```
$ curl -s https://storage.googleapis.com/evekit_md/2018/09/29/market_20180929.index.
↪gz | gzcat | head -n 10
market_18_20180929.history.gz 0
```

(continues on next page)

(continued from previous page)

```
market_19_20180929.history.gz 808
market_20_20180929.history.gz 1302
market_21_20180929.history.gz 2631
market_22_20180929.history.gz 3108
market_34_20180929.history.gz 3489
market_35_20180929.history.gz 6430
market_36_20180929.history.gz 9295
market_37_20180929.history.gz 12195
market_38_20180929.history.gz 14936
```

Suppose, for example, we would like to read market history for market type 34 (Tritanium). To read this data, we need to read data starting from offset 3489 and ending at offset 6429 (one less than the offset for market type 35). We can perform this extraction using a range request via curl. Note that the data is gzip compressed in the bulk file, so we need to decompress to read the results:

```
$ curl -s -H "range: bytes=3489-6429" https://storage.googleapis.com/evekit_md/2018/
↪09/29/market_20180929.bulk | gzcat | head -n 10
34,10000025,3,5.30,5.30,5.30,497677,1538179200000
34,10000028,25,3.80,3.80,3.80,140731594,1538179200000
34,10000029,102,6.00,6.00,6.00,196253830,1538179200000
34,10000030,259,5.21,5.37,5.31,316489920,1538179200000
34,10000031,8,4.99,4.99,4.99,49742860,1538179200000
34,10000016,438,3.58,5.00,4.35,53128283,1538179200000
34,10000020,187,4.50,4.87,4.67,66906196,1538179200000
34,10000022,2,0.10,0.10,0.10,1303633,1538179200000
34,10000023,3,6.00,6.00,6.00,30002,1538179200000
34,10000009,2,6.00,6.00,6.00,1029740,1538179200000
```

Note that you can read the entire bulk data file by removing the "range" argument. Given the small size of the market data files, this may be a more appropriate approach if you require history for several different types.

10.2.1 CAUTION: Reading Bulk Files with Zlib/Node.js

Bulk files are created by compressing and then concatenating the individual files stored in the compressed archive (e.g. the ".tgz" file). If you read a bulk file from start to finish you are therefore reading the concatenation of several individually compressed files. The standard GZip libraries are more than happy to handle a stream of concatenated gzip files. At time of writing, however, the standard ZLib library in Node.js does not handle a stream properly and will always stop reading at the end of the first compressed file. To work around this limitation, you'll either need to use a different compression library, or partition the data stream and read each partition individually. One way to partition the stream is to look for the GZip magic header bytes. This is the approach taken in the EveKit Market Data Server when requesting historic data. The code for splitting the buffer can be found here.

10.3 Order Books

Order book data records the set of buy and sell orders for a given type in a given region at a given time. The ESI refreshes order book data every five minutes. For a given day, there are therefore 288 snapshots for each type in each region. As with market history, an order book archive consists of a collection of files, one file per market type ID. Each file consists of 288 market snapshots for all regions where order book data was available for a given type. A sample listing for a recent file looks as follows:

```
$ tar tzf interval_20180929_5.tgz | head -n 10
interval_18_20180929_5.book.gz
```

(continues on next page)

(continued from previous page)

```
interval_19_20180929_5.book.gz
interval_20_20180929_5.book.gz
interval_21_20180929_5.book.gz
interval_22_20180929_5.book.gz
interval_34_20180929_5.book.gz
interval_35_20180929_5.book.gz
interval_36_20180929_5.book.gz
interval_37_20180929_5.book.gz
interval_38_20180929_5.book.gz
```

The format for order book files is slightly more complicated:

```
TYPE
SNAPSHOTS_PER_REGION
FIRST_REGION_ID
FIRST_REGION_FIRST_SNAPSHOT_TIME
FIRST_REGION_FIRST_SNAPSHOT_BUY_ORDER_COUNT
FIRST_REGION_FIRST_SNAPSHOT_SELL_ORDER_COUNT
FIRST_REGION_FIRST_SNAPSHOT_BUY_ORDER
...
FIRST_REGION_FIRST_SNAPSHOT_SELL_ORDER
...
FIRST_REGION_SECOND_SNAPSHOT_TIME
...
SECOND_REGION_ID
...
```

The first example in the listing above appears as follows:

```
$ gzcat interval_18_20180929_5.book.gz | head -n 10
18
288
10000025
1538179200000
0
0
1538179500000
0
0
1538179800000
```

In this example, the market type is Plagioclase (18) and each region will have 288 snapshots. The first region is Immensea (10000025) and the first snapshot is at 20180929 0000 UTC. There were no buy or sell orders at this time for this type in this region. The remaining snapshots will be sampled at five minute intervals. The next snapshot is therefore at 20180929 0005 UTC and likewise has no buy or sell orders.

When buy or sell orders occur, they are recorded in CSV format with the following fields:

Field	Description
ORDER ID	Unique market order ID.
BUY	"true" for buy orders, "false" otherwise.
ISSUED	Order issue date in milliseconds TUC (since the epoch).
PRICE	Order price.
VOLUME ENTERED	Volume entered when order was created.
MINIMUM VOLUME	Minimum volume requied for each order fill.
VOLUME	Current remaining volume to be filled in the order.
ORDER RANGE	Order range string. One of "station", "solarsystem", "region" or a number representing the number of jumps allowed from the station where the order was placed.
LOCATION ID	Location ID of station where order was entered.
DURATION	Order duration in days.

As an example, consider Tritanium (market type 34) from the example above:

```
$ gzcat interval_34_20180929_5.book.gz | head -n 10
34
288
10000025
1538179200000
3
0
5245221025,true,1536237612000,5.48,40000000,1,18803539,1,1027496149370,90
5252251352,true,1538069032000,4.15,25000000,1,3313490,1,1027011062756,90
5255545769,true,1537577368000,4.02,10000000,1,10000000,5,1027011062756,90
1538179500000
```

In this example, there were three buy orders in the first snapshot for Immensea. For convenience, **buy orders are always sorted highest price first**. Likewise, **sell orders are always sorted lowest price first**.

With 288 snapshots across approximately 8600 market types and 100 regions, order book files are significantly larger than market history. In most cases, it will be more appropriate to extract needed data from an online bulk file instead of extracting the entire archive.

Data can be extracted from order book bulk files using the same technique as for market history bulk files. First, the index file is consulted to discover the appropriate offsets for the requested data. Here are the first few lines from our example date:

```
curl -s https://storage.googleapis.com/evekit_md/2018/09/29/interval_20180929_5.index.
↪gz | gzcat | head -n 10
interval_18_20180929_5.book.gz 0
interval_19_20180929_5.book.gz 93941
interval_20_20180929_5.book.gz 146174
interval_21_20180929_5.book.gz 289411
interval_22_20180929_5.book.gz 343002
interval_34_20180929_5.book.gz 404514
interval_35_20180929_5.book.gz 737280
interval_36_20180929_5.book.gz 985574
interval_37_20180929_5.book.gz 1252779
interval_38_20180929_5.book.gz 1495309
```

A range request will then extract the requested data. As with market history, each range is gzip compressed and must be decompressed in order to read the results. The following example retrieves Tritanium from the online bulk file:

```
curl -s -H "range: bytes=404514-737279" https://storage.googleapis.com/evekit_md/2018/
↪09/29/interval_20180929_5.bulk | gzcat | head -n 10
34
288
10000025
1538179200000
3
0
5245221025,true,1536237612000,5.48,40000000,1,18803539,1,1027496149370,90
5252251352,true,1538069032000,4.15,25000000,1,3313490,1,1027011062756,90
5255545769,true,1537577368000,4.02,10000000,1,10000000,5,1027011062756,90
1538179500000
```

10.4 Inferred Trades

Inferred trade data is an estimation of trades that occurred in a given region on a given data. The ESI does not provide this data. Instead, we estimate likely trades by inspecting changes in the order book. The process by which trades are estimated is documented here.

All trades are collected into a single file per day. The format is similar to the order book format:

```
TYPE_ID
NUMBER_OF_REGIONS
REGION_1
TRADE_COUNT
TRADE_1
...
REGION_2
TRADE_COUNT
...
TYPE_ID
NUMBER_OF_REGIONS
...
```

The list of trades per region is in CSV format with the following fields:

Field	Description
TIMESTAMP	Trade timestamp in milliseconds UTC (since the epoch).
ACTUAL	"True" if this represents an actual trade, "False" if this trade is inferred.
BUY	"True" if this trade was a buy from an existing sell order, otherwise this trade was a sell to an existing buy order.
MATCHED ORDER ID	The order ID from which the buy or sell occurred.
TRADE PRICE	Trade price.
TRADE VOLUME	Trade volume.
LOCATION	The location ID of the station where the trade occurred, or "None" if it was not possible to determine location.

The following example shows the contents of the trade file for our example date:

```
$ gzcat trades_allregions_20180929.gz | head -n 10
18
14
```

(continues on next page)

(continued from previous page)

```
10000002
82
1538180400000,True,True,5259222306,58.42,28570,None
1538180700000,True,True,5259222306,58.42,46,None
1538180700000,True,True,5257674674,50.03,85,None
1538182200000,True,True,5259222306,58.42,22,None
1538183100000,True,True,5259222306,58.42,8847,None
1538188200000,True,True,5259222306,58.42,102,None
```

In this example, the first set of trades are reported for Plagioclase (market type 18). Trades were inferred for 14 regions on this date. The first region reported is "The Forge" which has 82 total trades. The first trade is an actual buy reported at 0020 UTC. The trade was for 28570 units at a price of 58.42 ISK. A location could not be inferred. This indicates that the matched sell order had range greater than "station".