django-eveuniverse

Erik Kalkoken

Jan 04, 2022

CONTENTS:

django-eveuniverse is a foundation app meant to help speed up the development of Eve Online apps with Django and ESI. It provides all classic "static" Eve classes as Django models, including all relationships, ready to be used in your project.

Furthermore, all Eve models have an on-demand loading mechanism for fetching new objects from ESI.

CHAPTER

ONE

OPERATIONS GUIDE

The operations guide describes how to install, configure and maintain *django-eveuniverse*.

1.1 Installation

To install django-eveuniverse into your Django project please follow the these steps:

1.1.1 Install from PyPI

You can install this app directly from PyPI:

```
pip install django-eveuniverse
```

1.1.2 Update settings

Next add `eveuniverse` to `INSTALLED_APPS` in your project's settings file.

By default only the core models are automatically loaded on-demand. If you want to also include some of the additional models please add them to your settings. (See also *settings* for a list of all available settings)

1.1.3 Setup celery

This app uses Celery for loading large sets of data, e.g. with the load commands. Please make sure celery is setup and working for your Django project.

Note: Note on celery worker setup

For an efficient loading of large amounts of data from ESI we recommend a thread based setup of celery workers with at least 10 concurrent workers.

For example on our test system with 20 gevent threads the loading of the complete Eve Online map (with the command: **eveuniverse_load_data map**) consisting of all regions, constellation and solar systems took only about 15 minutes.

1.1.4 Finalize installation

```
python manage.py migrate
```

Finally restart your Django instance so your changes become effective.

1.2 Updating

Hint: Before updating please always check the Change Log for any special instructions on updating or important changes that might affect your project.

To update your installation first install the new version:

```
pip install -U django-eveuniverse
```

Then run Django migrations:

```
python manage.py migrate
```

And finally restart your Django instance so your changes become effective.

1.3 Settings

Here is a list of available settings for this app. They can be configured by adding them to your local Django settings file.

Note: All settings are optional and the app will use the documented default settings if they are not used.

Important: Many settings will enable the automatic loading of related models. For example: if you enable Planets, all related planet object are automatically loaded when updating a solar system. This will significantly increase load times of objects, so we recommend to only enable additional models that are functionally needed.

The preferred approach is load related model on demand with the `enabled_sections` feature. Please see *Load related models on-demand* for more details.

`EVEUNIVERSE_LOAD_ASTEROID_BELTS = False`
When true will automatically load astroid belts with every solar system.

`EVEUNIVERSE_LOAD_DOGMAS = False`
When true will automatically load dogma, e.g. with every type.

`EVEUNIVERSE_LOAD_GRAPHICS = False`
When true will automatically load graphics with every type.

`EVEUNIVERSE_LOAD_MARKET_GROUPS = False`
When true will automatically load market groups with every type.

`EVEUNIVERSE_LOAD_MOONS = False`
When true will automatically load moons be with every planet.

EVEUNIVERSE_LOAD_PLANETS = False
When true will automatically load planets with every solar system.

EVEUNIVERSE_LOAD_STARGATES = False
When true will automatically load stargates with every solar system.

EVEUNIVERSE_LOAD_STARS = False
When true will automatically load stars with every solar system.

EVEUNIVERSE_LOAD_STATIONS = False
When true will automatically load stations be with every solar system.

EVEUNIVERSE_LOAD_TYPE_MATERIALS = False
When true will automatically load type materials be with every type.

EVEUNIVERSE_TASKS_TIME_LIMIT = 7200
Global timeout for tasks in seconds to reduce task accumulation during outages.

EVEUNIVERSE_USE_EVESKINSERVER = True
When True a call to EveType.icon_url for a SKIN type will return a eveskinserver URL else it will return a generic SKIN icon.

1.4 Management commands

The following management commands are available:

1.4.1 eveuniverse_load_data

This command will load a complete set of data form ESI and store it locally. Useful to optimize performance or when you want to provide the user with drop-down lists. Available sets:

- **map**: All regions, constellations and solar systems
- **ships**: All ship types
- **structures**: All structures types

1.4.2 eveuniverse_purge_all

This command will purge ALL data of your models

1.4.3 eveuniverse_load_types

```
Loads large sets of types as specified from ESI into the local database. This
is a helper command meant to be called from other apps only.

positional arguments:
  app_name              Name of app this data is loaded for

optional arguments:
  -h, --help            show this help message and exit
  --category_id CATEGORY_ID
                        Eve category ID to be loaded excl. dogma
```

(continues on next page)

(continued from previous page)

```
  --category_id_with_dogma CATEGORY_ID_WITH_DOGMA
                        Eve category ID to be loaded incl. dogma
  --disable_esi_check   Disables checking that ESI is online
  --group_id GROUP_ID   Eve group ID to be loaded excl. dogma
  --group_id_with_dogma GROUP_ID_WITH_DOGMA
                        Eve group ID to be loaded incl. dogma
  --type_id TYPE_ID     Eve type ID to be loaded excl. dogma
  --type_id_with_dogma TYPE_ID_WITH_DOGMA
                        Eve type ID to be loaded incl. dogma
```

1.5 Database tools

On some DBMS like MySQL it is not possible to reset the database and remove all eveuniverse tables with the standard "migrate zero" command. The reason is that eveuniverse is using composite primary keys and Django seams to have problems dealing with that correctly, when trying to roll back migrations.

As workaround you will need remove all tables with SQL commands. To make this easier we are providing a SQL script that contains all commands to drop the tables. The process for "migrating to zero" is then as follows:

1. Run SQL script `drop_tables.sql` on your database
2. Run `python manage.py migrate eveuniverse zero --fake`

CHAPTER

TWO

DEVELOPER GUIDE

The developer guide describes how to develop apps with *django-eveuniverse*.

2.1 Models

django-eveuniverse provides you with ready-made Django models for all Eve Universe classes. These models can be used like any other Django model in queries or included as related models in your app's own models.

Note: The "Eve Universe" classes are the classes from the Universe category in ESI plus the related classes for dogma and market groups. The objects of those classes change rarely and most changes are just adding new objects (e.g. new types). They are therefore well suited to be stored and cached locally for a longer period of time.

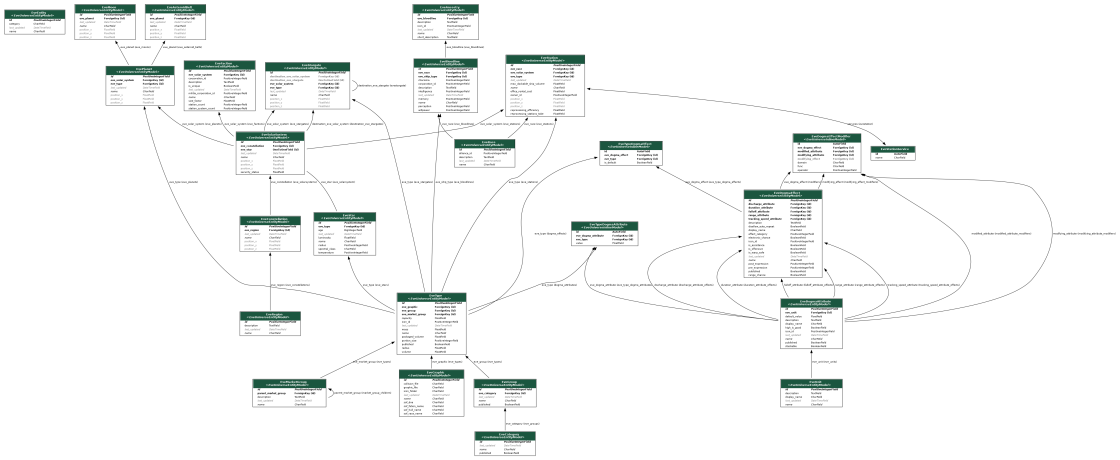
The Eve Universe classes consist mostly of the same objects as the Static Data Export (SDE).

See also:

Please see *Base classes* for the full documentation of all available models.

2.1.1 Relationship diagram

The following graph shows all models and how they are interrelated:



2.1.2 Naming of models and properties

The name of all models start with `Eve`. For example the model for solar systems is called `EveSolarSystem`.

All Eve model share the following basic properties:

- `id`: The Eve Online ID of the object and also the primary key
- `name`: The name of the eve objects
- `last_updated`: The date & time this object was last updated from ESI

Property names are mostly the same as in the ESI specification. The exceptions are:

- The common properties `id` and `name` as described above
- Boolean fields start with `is_`
- Properties that reference other Eve models always have the name of the referenced model, just in snake case. e.g. a property references a EveSolarSystem object would be called `eve_solar_system`.

2.1.3 Magic Methods

All Eve models have the following magic methods implemented:

- `__str__()`: returns the name of an object
- `__repr__()`: returns the complete object with all properties as model instance.

Examples:

```
>>> str(EveSolarSystem.objects.get(id=30000142))
'Jita'
```

```
>>> repr(EveSolarSystem.objects.get(id=30000142))
"EveSolarSystem(eve_constellation_id=20000020, eve_star_id=None, id=30000142, name='Jita
↪', position_x=-1.2906486173487826e+17, position_y=6.075530690996363e+16, position_z=1.
↪1746922706009029e+17, security_status=0.9459131360054016)"
```

2.1.4 Additional functionality in Eve Models

Some Eve models provide additional useful functionality, e.g. icon image URLs.

For example the `EveSolarSystem` comes with a lot of additional features incl. a route finder:

```
>>> jita = EveSolarSystem.objects.get(id=30000142)
>>> akidagi = EveSolarSystem.objects.get(id=30045342)
>>> jita.jumps_to(akidagi)
10
```

2.2 Fetching eve objects from ESI

2.2.1 Fetching eve objects on-demand

To fetch an eve object you can simply call it's manager method `get_or_create_esi()`. This will return the requested eve objects from the database if it exists, or else automatically load it from ESI:

For example for getting the solar system of Jita on-demand you could do the following:

```
>>> EveSolarSystem.objects.get_or_create_esi(id=30000142)
(EveSolarSystem(eve_constellation_id=20000020, eve_star_id=None, id=30000142, name='Jita
↪', position_x=-1.2906486173487826e+17, position_y=6.075530690996363e+16, position_z=1.
↪1746922706009029e+17, security_status=0.9459131360054016), True)
```

Or if want to fetch a fresh Eve object from ESI you can call the manager method `update_or_create_esi()`. This which will always retrieve a new Eve objects from ESI and update the local copy.

Our example for Jita would then look like this:

```
>>> EveSolarSystem.objects.get_or_create_esi(id=30000142)
(EveSolarSystem(eve_constellation_id=20000020, eve_star_id=None, id=30000142, name='Jita
↪', position_x=-1.2906486173487826e+17, position_y=6.075530690996363e+16, position_z=1.
↪1746922706009029e+17, security_status=0.9459131360054016), False)
```

Hint: Please see *Manager methods* for an overview of all available methods.

2.2.2 Fetching parent and child objects

Many Eve models have parent and child models. For example `EveSolarSystem` has `EveConstellation` as parent model, and `EvePlanet` is one of its child models. When fetching an Eve objects for the first time from ESI, the related parent objects will automatically be loaded to preserve the integrity of the database.

For example if you are fetching Jita for the first time, the objects for Jita's constellation (parent of solar system) and Jita's region (parent of constellation) will be fetched too.

In addition it is possible to automatically fetch all children of an object. This can be very useful for loading larger sets of data. For example, if you want to load all ship types, you can just fetch the inventory category for ships and include children by setting `include_children` to `True`.

Example:

```
>>> EveCategory.objects.get_or_create_esi(id=6, include_children=True)
(EveCategory(id=6, name='Ship', published=True), False)
```

This will load all children blocking, which can take quite some time. For large sets of data it is often is better to load children async (via Celery). This can be done by setting `wait_for_children` to `False`.

```
>>> EveCategory.objects.get_or_create_esi(id=6, include_children=True, wait_for_
↪children=False)
(EveCategory(id=6, name='Ship', published=True), False)
```

2.2.3 Selecting which related models are loaded

Eve models are heavily interrelated and trying to load just a small subset of objects can quickly cascade to loading large parts of the whole universe. However, not all of those related models are needed by every app and always loading them would only increase overall response times and fill up the database without adding much value for the app.

For example the "dogma" consists of 4 models that relate to inventory types and contain specifics for type objects like the rate of fire for some ship modules. Not every app may need that additional information in their database.

Therefore the following eve models are not loaded through relations By default and need to be turned on explicitly:

- EveAsteroidBelt
- EveDogmaAttribute
- EveDogmaEffect
- EveGraphic
- EveMarketGroup
- EveMoon
- EvePlanet
- EveStargate
- EveStar
- EveStation

Hint: You can still load objects from disabled models directly - e.g. with `get_or_create_esi()` - but be mindful that relations will not be created automatically, which can lead to inconsistencies in your database.

There are two solutions for loading disabled models incl. their relations:

- Globally enabling disabled models
- Enabling disabled models on-demand

Note: Related models that are disabled by default are also called sections.

Globally enabling models

One solution here is to offer developers control over which related models are loaded through configuration. Each disabled model therefore as a corresponding setting that can be used to globally enable that model.

Hint: When turning on loading of related models you usually want to reload related eve objects that already exist in the database to make sure all relations are created correctly. e.g. after turning on `EveStargate` you want to reload all solar systems.

See also:

For an overview of all settings please see *Settings*.

2.2.4 Load related models on-demand

However, globally enabling those related models will affect all apps of a Django installation. For instance if you turn on dogmas globally, dogmas will be loaded for each and every type, even if it that extra data is not needed.

This might not be the best option for some use cases and we are therefore offering an alternative solution. You can also activate disabled models on-demand.

Manager methods

Most manager methods like `get_or_create_esi()` and `update_or_create_esi()` take an extra argument called `enabled_sections`, which allows you to ensure specific sections are loaded with your query.

For example you may want to load planets just for one solar system. Here is how such a request might look like. Note that eveuniverse will automatically load missing planets from ESI for that solar system, even if it already exist in the local database.

```
obj, _ = EveSolarSystem.objects.get_or_create_esi(id=30000142, include_children=True,␣
↪enabled_sections=[EveSolarSystem.Section.PLANETS])
```

You can also specify multiple sections with one request. Here is how to fetch planets and their respective moons (a section of EvePlanet) on demand:

```
obj, _ = EveSolarSystem.objects.get_or_create_esi(id=30000142, include_children=True,␣
↪enabled_sections=[EveSolarSystem.Section.PLANETS, EvePlant.Section.MOONS])
```

See also:

See also the API for a list of all available sections for each model that supports it: *eveuniverse.models.EvePlanet.Section*, *eveuniverse.models.EveSolarSystem.Section*, *eveuniverse.models.EveType.Section*.

Preloading instances

`eveuniverse_load_types` management command for preloading types can also include loading dogmas if requested.

Test tools

The test tool for creating *test data* also support the `enabled_sections` argument.

2.2.5 Preloading data

While all models support loading eve objects on demand from ESI, some apps might need specific data sets to be preloaded. For example an app might want to provide a drop down list of all structure types, and loading that list on demand would not be fast enough to guarantee acceptable UI response times.

The solution is to provide the user with a management command, so he an preload the needed data sets - for example all ship types - during app installation. Since this is a command use case *django-eveuniverse* offers a management helper command with all the needed functionality for loading data and which can be easily utilized with just a very small and simple management command in your own app.

Here is an example for creating such a management command. We want to load all kinds of structures to show to the user in a drop down list. We therefore want to preload all structure types (`category_id = 65`), all control towers (`group_id = 365`) and the customs office (`type_id = 2233`):

```
from django.core.management import call_command
from django.core.management.base import BaseCommand


class Command(BaseCommand):
    help = "Preloads data required for this app from ESI"

    def handle(self, *args, **options):
        call_command(
            "eveuniverse_load_types",
            __title__,
            "--category_id",
            "65",
            "--group_id",
            "365",
            "--type_id",
            "2233",
        )
```

For more details on how to use `eveuniverse_load_types` just call it with `--help` from a console.

See also:

For an overview of all management commands please see *Management commands*.

2.3 Eve ID to name resolution

A common problem when working with data from ESI is the need to resolve the ID of an Eve object to it's name. To make this easier *django-eveuniverse* provides a dedicated model called `EveEntity`. `EveEntity` allows you to quickly resolve large amounts of Eve IDs to objects in bulk, with every object having it's name and entity category. Resolved objects are stored locally and automatically used to speed up subsequent ID resolving.

Here is a simple example for resolving the ID of the Jita solar system:

```
>>> EveEntity.objects.resolve_name(30000142)
'Jita'
```

Note: Eve IDs have unique ranged for the supported categories, which means they can be safely resolved without having to specify a category.

This examples show how to resolve a list of IDs in bulk and using a resolver object to access the results:

```
>>> resolver = EveEntity.objects.bulk_resolve_names([30000142, 34562, 498125261])
>>> resolver.to_name(30000142)
'Jita'
>>> resolver.to_name(34562)
'Svipul'
>>> resolver.to_name(498125261)
'Svipul'
>>> resolver.to_name(498125261)
'Test Alliance Please Ignore'
```

Another approach is to bulk create EveEntity objects with the ID only and then resolve all "new" objects with `EveEntity.objects.bulk_update_new_esi()`. This approach works well when using EveEntity objects as property in you app's models.

Hint: If you need to test that an ID is valid you can use `get_or_create_esi()` or `update_or_create_esi()`. Both will return `None` instead of an `EveEntity` object if the given ID was not valid. You can also use `resolve_name()`, which will return an empty string for invalid IDs.

However, calling ESI with an invalid ID will also count against the error rate limit, so use with care.

See also:

For more features and details please see *EveEntity manager methods*.

2.4 Test data

django-eveuniverse comes with tools that help you generate and use test data for your own apps.

2.4.1 Generate test data

To generate your test data create a script within your projects and run that scrip as a Django test. That is important to ensure that the database on which the scripts operates is empty. That script will then create a JSON file that contains freshly retrieved Eve objects from ESI based on your specification.

create_eveuniverse.py

Here is an example script for generating test data (taken from aa-killtracker):

```
from django.test import TestCase

from eveuniverse.tools.testdata import create_testdata, ModelSpec

from . import test_data_filename


class CreateEveUniverseTestData(TestCase):
    def test_create_testdata(self):
        testdata_spec = [
            ModelSpec("EveFaction", ids=[500001]),
            ModelSpec(
                "EveType",
                ids=[603, 621, 638, 2488, 2977, 3756, 11379, 16238, 34562, 37483],
            ),
            ModelSpec(
                "EveSolarSystem", ids=[30001161, 30004976, 30004984, 30045349, 31000005],
            ),
            ModelSpec("EveRegion", ids=[10000038], include_children=True),
        ]
        create_testdata(testdata_spec, test_data_filename())
```

2.4.2 Using generated testdata in your tests

To utilize the generated testdata file in your test you need another script that creates objects from your generated JSON file.

load_eveuniverse.py

Here is an example script that creates objects from the JSON file.

```
import json

from eveuniverse.tools.testdata import load_testdata_from_dict

from . import test_data_filename


def _load_eveuniverse_from_file():
    with open(test_data_filename(), "r", encoding="utf-8") as f:
        data = json.load(f)

    return data


eveuniverse_testdata = _load_eveuniverse_from_file()


def load_eveuniverse():
    load_testdata_from_dict(eveuniverse_testdata)
```

You can then load all Eve objects in your own test script like so:

test_example.py

```
from django.test import TestCase
from .load_eveuniverse import load_eveuniverse

class MyTest(TestCase):

  @classmethod
  def setUpClass(cls):
      super().setUpClass()
      load_eveuniverse()

  def test_my_test(self):
    svipul = EveType.objects.get(id=34562)
    # ...
```

CHAPTER

THREE

API

This chapter contains the developer reference documentation of the public API for *django-eveuniverse* consisting of all models, their manager methods and helpers.

3.1 Base classes

class EveUniverseBaseModel(**args*, ***kwargs*)
Base class for all Eve Universe Models

classmethod all_models() → List[Dict[django.db.models.base.Model, int]]
returns a list of all Eve Universe model classes sorted by load order

classmethod get_model_class(*model_name: str*) → django.db.models.base.Model
returns the model class for the given name

class EveUniverseEntityModel(**args*, ***kwargs*)
Base class for Eve Universe Entity models

Entity models are normal Eve entities that have a dedicated ESI endpoint

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI

class Section(*value*)
An enumeration.

classmethod eve_entity_category() → str
returns the EveEntity category of this model if one exists else and empty string

3.2 Core functions

3.2.1 dotlan

Generates profile URLs for dotlan.

alliance_url(*name: str*) → str
URL for page about given alliance on dotlan.

corporation_url(*name: str*) → str
URL for page about given corporation on dotlan.

faction_url(*name: str*) → str
URL for page about given corporation on dotlan.

region_url(*name: str*) → str
URL for page about given region on dotlan.

solar_system_url(*name: str*) → str
URL for page about given solar system on dotlan.

3.2.2 esitools

Tools for interacting with ESI.

is_esi_online() → bool
Checks if the Eve servers are online. Returns True if there are, else False

3.2.3 eveimageserver

Fetching image URLs for Eve objects.

class EsiCategory(*value*)
Enum class for ESI categories

class EsiTenant(*value*)
Enum class for ESI tenants

class ImageVariant(*value*)
Enum class for image variants

alliance_logo_url(*alliance_id: int*, *size: int = 32*) → str
image URL for the given alliance ID

character_portrait_url(*character_id: int*, *size: int = 32*) → str
image URL for the given character ID

corporation_logo_url(*corporation_id: int*, *size: int = 32*) → str
image URL for the given corporation ID

faction_logo_url(*faction_id: int*, *size: int = 32*) → str
image URL for the given alliance ID

type_bp_url(*type_id: int*, *size: int = 32*) → str
blueprint original image URL for the given type ID

type_bpc_url(*type_id: int*, *size: int = 32*) → str
blueprint copy image URL for the given type ID

type_icon_url(*type_id: int*, *size: int = 32*) → str
icon image URL for the given type ID

type_render_url(*type_id: int*, *size: int = 32*) → str
render image URL for the given type ID

3.2.4 eveitems

URLs for profile pages on eveitems webpage.

type_url(*type_id: int*) → str
URL to display any type on the default third party webpage.

3.2.5 evemicros

Wrapper to access evemicros API.

class EveItem(*id*, *name*, *type_id*, *distance*)

property distance
Alias for field number 3

property id
Alias for field number 0

property name
Alias for field number 1

property type_id
Alias for field number 2

nearest_celestial(*solar_system_id: int*, *x: int*, *y: int*, *z: int*, *group_id: Optional[int] = None*) → Optional[*eveuniverse.core.evemicros.EveItem*]
Fetch nearest celestial to given coordinates from API. Results are cached.

Parameters

- **solar_system_id** – Eve ID of solar system
- **x** – Start point in space to look from
- **y** – Start point in space to look from
- **z** – Start point in space to look from
- **group_id** – Eve ID of group to filter results by

Raises **HTTPError** – If an HTTP error is encountered

Returns Found Eve item or None if nothing found nearby.

3.2.6 eveskinserver

Wrapper for accessing eveskinserver API.

type_icon_url(*type_id: int*, *size: int = 32*) → str
icon image URL for the given SKIN type ID

3.2.7 evewho

Wrapper for generating profile URLs on evewho.

alliance_url(*eve_id: int*) → str
URL for page about given alliance on evewho.

character_url(*eve_id: int*) → str
URL for page about given character on evewho.

corporation_url(*eve_id: int*) → str
URL for page about given corporation on evewho.

3.3 Eve Models

3.3.1 EveAncestry

class EveAncestry(**args*, ***kwargs*)
An ancestry in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **eve_bloodline** (ForeignKey to *EveBloodline*) – Eve bloodline
- **description** (*TextField*) – Description
- **icon_id** (*PositiveIntegerField*) – Icon id
- **short_description** (*TextField*) – Short description

exception DoesNotExist

exception MultipleObjectsReturned

3.3.2 EveAsteroidBelt

class EveAsteroidBelt(**args*, ***kwargs*)
An asteroid belt in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **eve_planet** (ForeignKey to *EvePlanet*) – Eve planet
- **position_x** (*FloatField*) – Position x. x position in the solar system
- **position_y** (*FloatField*) – Position y. y position in the solar system
- **position_z** (*FloatField*) – Position z. z position in the solar system

exception DoesNotExist

exception MultipleObjectsReturned

3.3.3 EveBloodline

class EveBloodline(**args*, ***kwargs*)
A bloodline in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **eve_race** (ForeignKey to *EveRace*) – Eve race
- **eve_ship_type** (ForeignKey to *EveType*) – Eve ship type
- **charisma** (*PositiveIntegerField*) – Charisma
- **corporation_id** (*PositiveIntegerField*) – Corporation id
- **description** (*TextField*) – Description
- **intelligence** (*PositiveIntegerField*) – Intelligence
- **memory** (*PositiveIntegerField*) – Memory
- **perception** (*PositiveIntegerField*) – Perception
- **willpower** (*PositiveIntegerField*) – Willpower

exception DoesNotExist

exception MultipleObjectsReturned

3.3.4 EveCategory

class EveCategory(**args*, ***kwargs*)
An inventory category in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **published** (*BooleanField*) – Published

exception DoesNotExist

exception MultipleObjectsReturned

3.3.5 EveConstellation

class EveConstellation(**args*, ***kwargs*)
A star constellation in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **eve_region** (ForeignKey to *EveRegion*) – Eve region
- **position_x** (*FloatField*) – Position x. x position in the solar system
- **position_y** (*FloatField*) – Position y. y position in the solar system
- **position_z** (*FloatField*) – Position z. z position in the solar system

exception DoesNotExist

exception MultipleObjectsReturned

classmethod eve_entity_category() → str
returns the EveEntity category of this model if one exists else and empty string

3.3.6 EveDogmaAttribute

class EveDogmaAttribute(**args*, ***kwargs*)
A dogma attribute in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI

- **eve_unit** (ForeignKey to *EveUnit*) – Eve unit
- **default_value** (*FloatField*) – Default value
- **description** (*TextField*) – Description
- **display_name** (*CharField*) – Display name
- **high_is_good** (*BooleanField*) – High is good
- **icon_id** (*PositiveIntegerField*) – Icon id
- **published** (*BooleanField*) – Published
- **stackable** (*BooleanField*) – Stackable

exception DoesNotExist

exception MultipleObjectsReturned

3.3.7 EveDogmaEffect

class EveDogmaEffect(**args*, ***kwargs*)

A dogma effect in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **name** (*CharField*) – Name. Eve Online name
- **description** (*TextField*) – Description
- **disallow_auto_repeat** (*BooleanField*) – Disallow auto repeat
- **discharge_attribute** (ForeignKey to *EveDogmaAttribute*) – Discharge attribute
- **display_name** (*CharField*) – Display name
- **duration_attribute** (ForeignKey to *EveDogmaAttribute*) – Duration attribute
- **effect_category** (*PositiveIntegerField*) – Effect category
- **electronic_chance** (*BooleanField*) – Electronic chance
- **falloff_attribute** (ForeignKey to *EveDogmaAttribute*) – Falloff attribute
- **icon_id** (*PositiveIntegerField*) – Icon id
- **is_assistance** (*BooleanField*) – Is assistance
- **is_offensive** (*BooleanField*) – Is offensive
- **is_warp_safe** (*BooleanField*) – Is warp safe
- **post_expression** (*PositiveIntegerField*) – Post expression
- **pre_expression** (*PositiveIntegerField*) – Pre expression
- **published** (*BooleanField*) – Published
- **range_attribute** (ForeignKey to *EveDogmaAttribute*) – Range attribute
- **range_chance** (*BooleanField*) – Range chance

- **tracking_speed_attribute** (ForeignKey to *EveDogmaAttribute*) – Tracking speed attribute

exception **DoesNotExist**

exception **MultipleObjectsReturned**

3.3.8 EveDogmaEffectModifier

class **EveDogmaEffectModifier**(**args*, ***kwargs*)

A modifier for a dogma effect in Eve Online

Parameters

- **id** (*AutoField*) – Id
- **domain** (*CharField*) – Domain
- **eve_dogma_effect** (ForeignKey to *EveDogmaEffect*) – Eve dogma effect
- **func** (*CharField*) – Func
- **modified_attribute** (ForeignKey to *EveDogmaAttribute*) – Modified attribute
- **modifying_attribute** (ForeignKey to *EveDogmaAttribute*) – Modifying attribute
- **modifying_effect** (ForeignKey to *EveDogmaEffect*) – Modifying effect
- **operator** (*PositiveIntegerField*) – Operator

exception **DoesNotExist**

exception **MultipleObjectsReturned**

3.3.9 Eve Entity

class **EveEntity**(**args*, ***kwargs*)

An Eve object from one of the categories supported by ESI's */universe/names/* endpoint:

alliance, character, constellation, faction, type, region, solar system, station

This is a special model model dedicated to quick resolution of Eve IDs to names and their categories, e.g. for characters. See also manager methods.

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **category** (*CharField*) – Category

__init__(**args*, ***kwargs*) → None

icon_url(*size: int = 64*) → str

Create image URL for related EVE icon

Parameters **size** – size of image file in pixels, allowed values: 32, 64, 128, 256, 512

Returns strings with image URL

property is_alliance: bool
returns True if entity is an alliance, else False

is_category(*category: str*) → bool
returns True if this entity has the given category, else False

property is_character: bool
returns True if entity is a character, else False

property is_constellation: bool
returns True if entity is a constellation, else False

property is_corporation: bool
returns True if entity is a corporation, else False

property is_faction: bool
returns True if entity is a faction, else False

property is_npc: bool
returns True if this entity is an NPC character or NPC corporation, else False

property is_region: bool
returns True if entity is a region, else False

property is_solar_system: bool
returns True if entity is a solar system, else False

property is_station: bool
returns True if entity is a station, else False

property is_type: bool
returns True if entity is an inventory type, else False

property profile_url: str
URL to default third party website with profile info about this entity.

Supported for: alliance, character, corporation, faction, region, solar system, type

update_from_esi() → *eveuniverse.models.EveEntity*
Update the current object from ESI

Returns itself after update

3.3.10 EveFaction

class EveFaction(**args*, ***kwargs*)
A faction in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **corporation_id** (*PositiveIntegerField*) – Corporation id
- **description** (*TextField*) – Description
- **eve_solar_system** (ForeignKey to *EveSolarSystem*) – Eve solar system
- **is_unique** (*BooleanField*) – Is unique

- **militia_corporation_id** (*PositiveIntegerField*) – Militia corporation id
- **size_factor** (*FloatField*) – Size factor
- **station_count** (*PositiveIntegerField*) – Station count
- **station_system_count** (*PositiveIntegerField*) – Station system count

exception DoesNotExist

exception MultipleObjectsReturned

classmethod eve_entity_category() → str
returns the EveEntity category of this model if one exists else and empty string

logo_url(*size=64*) → str
returns an image URL for this faction

Parameters **size** – optional size of the image

property profile_url: str
URL to default third party website with profile info about this entity.

3.3.11 EveGraphic

class EveGraphic(**args*, ***kwargs*)
A graphic in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **collision_file** (*CharField*) – Collision file
- **graphic_file** (*CharField*) – Graphic file
- **icon_folder** (*CharField*) – Icon folder
- **sof_dna** (*CharField*) – Sof dna
- **sof_fation_name** (*CharField*) – Sof fation name
- **sof_hull_name** (*CharField*) – Sof hull name
- **sof_race_name** (*CharField*) – Sof race name

exception DoesNotExist

exception MultipleObjectsReturned

3.3.12 EveGroup

class EveGroup(**args*, ***kwargs*)
An inventory group in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **eve_category** (ForeignKey to *EveCategory*) – Eve category
- **published** (*BooleanField*) – Published

exception DoesNotExist

exception MultipleObjectsReturned

3.3.13 EveMarketGroup

class EveMarketGroup(**args*, ***kwargs*)
A market group in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **description** (*TextField*) – Description
- **parent_market_group** (ForeignKey to *EveMarketGroup*) – Parent market group

exception DoesNotExist

exception MultipleObjectsReturned

3.3.14 EveMarketPrice

class EveMarketPrice(**args*, ***kwargs*)
A market price of an Eve Online type

Parameters

- **eve_type** (OneToOneField to *EveType*) – Eve type
- **adjusted_price** (*FloatField*) – Adjusted price
- **average_price** (*FloatField*) – Average price
- **updated_at** (*DateTimeField*) – Updated at

exception DoesNotExist

exception MultipleObjectsReturned

3.3.15 EveMoon

class EveMoon(**args*, ***kwargs*)
A moon in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **eve_planet** (ForeignKey to *EvePlanet*) – Eve planet
- **position_x** (*FloatField*) – Position x. x position in the solar system
- **position_y** (*FloatField*) – Position y. y position in the solar system
- **position_z** (*FloatField*) – Position z. z position in the solar system

exception DoesNotExist

exception MultipleObjectsReturned

3.3.16 EvePlanet

class EvePlanet(**args*, ***kwargs*)
A planet in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **eve_solar_system** (ForeignKey to *EveSolarSystem*) – Eve solar system
- **eve_type** (ForeignKey to *EveType*) – Eve type
- **position_x** (*FloatField*) – Position x. x position in the solar system
- **position_y** (*FloatField*) – Position y. y position in the solar system
- **position_z** (*FloatField*) – Position z. z position in the solar system
- **enabled_sections** (*BitField*) – Enabled sections. Flags for loadable sections. True if instance was loaded with section.

exception DoesNotExist

exception MultipleObjectsReturned

class Section(*value*)
Sections that can be optionally loaded with each instance

ASTEROID_BELTS = 'asteroid_belts'

MOONS = 'moons'

3.3.17 EveRace

class EveRace(**args*, ***kwargs*)
A race in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **alliance_id** (*PositiveIntegerField*) – Alliance id
- **description** (*TextField*) – Description

exception DoesNotExist

exception MultipleObjectsReturned

3.3.18 EveRegion

class EveRegion(**args*, ***kwargs*)
A star region in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **description** (*TextField*) – Description

exception DoesNotExist

exception MultipleObjectsReturned

classmethod eve_entity_category() → str
returns the EveEntity category of this model if one exists else and empty string

property profile_url: str
URL to default third party website with profile info about this entity.

3.3.19 EveSolarSystem

class EveSolarSystem(**args*, ***kwargs*)
A solar system in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **eve_constellation** (ForeignKey to *EveConstellation*) – Eve constellation

- **eve_star** (OneToOneField to *EveStar*) – Eve star
- **position_x** (*FloatField*) – Position x. x position in the solar system
- **position_y** (*FloatField*) – Position y. y position in the solar system
- **position_z** (*FloatField*) – Position z. z position in the solar system
- **security_status** (*FloatField*) – Security status
- **enabled_sections** (*BitField*) – Enabled sections. Flags for loadable sections. True if instance was loaded with section.

exception DoesNotExist

exception MultipleObjectsReturned

class NearestCelestial(*eve_type*, *eve_object*, *distance*)
Container for a nearest celestial

property distance
Alias for field number 2

property eve_object
Alias for field number 1

property eve_type
Alias for field number 0

class Section(*value*)
Sections that can be optionally loaded with each instance

PLANETS = 'planets'

STARGATES = 'stargates'

STARS = 'stars'

STATIONS = 'stations'

distance_to(*destination:* eveuniverse.models.EveSolarSystem) → Optional[float]
Calculates the distance in meters between the current and the given solar system

Parameters **destination** – Other solar system to use in calculation

Returns Distance in meters or None if one of the systems is in WH space

classmethod eve_entity_category() → str
returns the EveEntity category of this model if one exists else and empty string

property is_high_sec: bool
returns True if this solar system is in high sec, else False

property is_low_sec: bool
returns True if this solar system is in low sec, else False

property is_null_sec: bool
returns True if this solar system is in null sec, else False

property is_w_space: bool
returns True if this solar system is in wormhole space, else False

jumps_to(*destination:* eveuniverse.models.EveSolarSystem) → Optional[int]
Calculates the shortest route between the current and the given solar system

Parameters **destination** – Other solar system to use in calculation

Returns Number of total jumps or None if no route can be found (e.g. if one system is in WH space)

nearest_celestial(*x: int*, *y: int*, *z: int*, *group_id: Optional[int] = None*) → Optional[*eveuniverse.models.NearestCelestial*]

Determine nearest celestial to given coordinates as eveuniverse object.

Parameters

- **x** – Start point in space to look from
- **y** – Start point in space to look from
- **z** – Start point in space to look from
- **group_id** – Eve ID of group to filter results by

Raises **HTTPError** – If an HTTP error is encountered

Returns Eve item or None if none is found

property profile_url: str

URL to default third party website with profile info about this entity.

route_to(*destination:* eveuniverse.models.EveSolarSystem) → Optional[List[*eveuniverse.models.EveSolarSystem*]]

Calculates the shortest route between the current and the given solar system

Parameters **destination** – Other solar system to use in calculation

Returns List of solar system objects incl. origin and destination or None if no route can be found (e.g. if one system is in WH space)

3.3.20 EveStar

class EveStar(**args*, ***kwargs*)

A star in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **age** (*BigIntegerField*) – Age
- **eve_type** (ForeignKey to *EveType*) – Eve type
- **luminosity** (*FloatField*) – Luminosity
- **radius** (*PositiveIntegerField*) – Radius
- **spectral_class** (*CharField*) – Spectral class
- **temperature** (*PositiveIntegerField*) – Temperature

exception DoesNotExist

exception MultipleObjectsReturned

3.3.21 EveStargate

class EveStargate(**args*, ***kwargs*)
A stargate in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **destination_eve_stargate** (OneToOneField to *EveStargate*) – Destination eve stargate
- **destination_eve_solar_system** (ForeignKey to *EveSolarSystem*) – Destination eve solar system
- **eve_solar_system** (ForeignKey to *EveSolarSystem*) – Eve solar system
- **eve_type** (ForeignKey to *EveType*) – Eve type
- **position_x** (*FloatField*) – Position x. x position in the solar system
- **position_y** (*FloatField*) – Position y. y position in the solar system
- **position_z** (*FloatField*) – Position z. z position in the solar system

exception DoesNotExist

exception MultipleObjectsReturned

3.3.22 EveStation

class EveStation(**args*, ***kwargs*)
A space station in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **eve_race** (ForeignKey to *EveRace*) – Eve race
- **eve_solar_system** (ForeignKey to *EveSolarSystem*) – Eve solar system
- **eve_type** (ForeignKey to *EveType*) – Eve type
- **max_dockable_ship_volume** (*FloatField*) – Max dockable ship volume
- **office_rental_cost** (*FloatField*) – Office rental cost
- **owner_id** (*PositiveIntegerField*) – Owner id
- **position_x** (*FloatField*) – Position x. x position in the solar system
- **position_y** (*FloatField*) – Position y. y position in the solar system
- **position_z** (*FloatField*) – Position z. z position in the solar system

- **reprocessing_efficiency** (*FloatField*) – Reprocessing efficiency
- **reprocessing_stations_take** (*FloatField*) – Reprocessing stations take
- **services** (*ManyToManyField*) – Services

exception DoesNotExist

exception MultipleObjectsReturned

classmethod eve_entity_category() → str
returns the EveEntity category of this model if one exists else and empty string

3.3.23 EveStationService

class EveStationService(*args, **kwargs)
A service in a space station

Parameters

- **id** (*AutoField*) – Id
- **name** (*CharField*) – Name

exception DoesNotExist

exception MultipleObjectsReturned

3.3.24 EveType

class EveType(*args, **kwargs)
An inventory type in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **capacity** (*FloatField*) – Capacity
- **eve_group** (ForeignKey to *EveGroup*) – Eve group
- **eve_graphic** (ForeignKey to *EveGraphic*) – Eve graphic
- **icon_id** (*PositiveIntegerField*) – Icon id
- **eve_market_group** (ForeignKey to *EveMarketGroup*) – Eve market group
- **mass** (*FloatField*) – Mass
- **packaged_volume** (*FloatField*) – Packaged volume
- **portion_size** (*PositiveIntegerField*) – Portion size
- **radius** (*FloatField*) – Radius
- **published** (*BooleanField*) – Published
- **volume** (*FloatField*) – Volume

- **enabled_sections** (*BitField*) – Enabled sections. Flags for loadable sections. True if instance was loaded with section.

exception DoesNotExist

class IconVariant(*value*)
Variant of icon to produce with *icon_url()*

BPC = 3
blueprint copy

BPO = 2
blueprint original

REGULAR = 1
anything, except blueprint or skin

SKIN = 4
SKIN

exception MultipleObjectsReturned

class Section(*value*)
Sections that can be optionally loaded with each instance

DOGMAS = 'dogmas'

GRAPHICS = 'graphics'

MARKET_GROUPS = 'market_groups'

TYPE_MATERIALS = 'type_materials'

classmethod eve_entity_category() → str
returns the EveEntity category of this model if one exists else and empty string

icon_url(*size: int = 64*, *variant: Optional[*eveuniverse.models.EveType.IconVariant*] = None*, *category_id: Optional[int] = None*, *is_blueprint: Optional[bool] = None*) → str
returns an image URL to this type as icon. Also works for blueprints and SKINs.

Will try to auto-detect the variant based on the types's category, unless *variant* or *category_id* is specified.

Parameters

- **variant** – icon variant to use
- **category_id** – category ID of this type
- **is_blueprint** – DEPRECATED - type is assumed to be a blueprint

property profile_url: str
URL to display this type on the default third party webpage.

render_url(*size=64*) → str
return an image URL to this type as render

3.3.25 EveTypeDogmaAttribute

class EveTypeDogmaAttribute(**args*, ***kwargs*)
A dogma attribute of on inventory type in Eve Online

Parameters

- **id** (*AutoField*) – Id
- **eve_dogma_attribute** (ForeignKey to *EveDogmaAttribute*) – Eve dogma attribute
- **eve_type** (ForeignKey to *EveType*) – Eve type
- **value** (*FloatField*) – Value

exception DoesNotExist

exception MultipleObjectsReturned

3.3.26 EveTypeDogmaEffect

class EveTypeDogmaEffect(**args*, ***kwargs*)
A dogma effect of on inventory type in Eve Online

Parameters

- **id** (*AutoField*) – Id
- **eve_dogma_effect** (ForeignKey to *EveDogmaEffect*) – Eve dogma effect
- **eve_type** (ForeignKey to *EveType*) – Eve type
- **is_default** (*BooleanField*) – Is default

exception DoesNotExist

exception MultipleObjectsReturned

3.3.27 EveTypeMaterial

class EveTypeMaterial(**args*, ***kwargs*)
Material type for an Eve online type

Parameters

- **id** (*AutoField*) – Id
- **eve_type** (ForeignKey to *EveType*) – Eve type
- **material_eve_type** (ForeignKey to *EveType*) – Material eve type
- **quantity** (*PositiveIntegerField*) – Quantity

exception DoesNotExist

exception MultipleObjectsReturned

3.3.28 EveUnit

class EveUnit(**args*, ***kwargs*)

A unit in Eve Online

Parameters

- **id** (*PositiveIntegerField*) – Id. Eve Online ID
- **name** (*CharField*) – Name. Eve Online name
- **last_updated** (*DateTimeField*) – Last updated. When this object was last updated from ESI
- **display_name** (*CharField*) – Display name
- **description** (*TextField*) – Description

exception DoesNotExist

exception MultipleObjectsReturned

3.4 Manager methods

3.4.1 Default manager methods

All eve models have the following manager methods:

class EveUniverseEntityModelManager(**args*, ***kwargs*)

bulk_get_or_create_esi(***, *ids: List[int]*, *include_children: bool = False*, *wait_for_children: bool = True*, *enabled_sections: Optional[Iterable[str]] = None*) → django.db.models.query.QuerySet

Gets or creates objects in bulk.

Nonexisting objects will be fetched from ESI (blocking). Will always get/create parent objects.

Parameters

- **ids** – List of valid IDs of Eve objects
- **include_children** – when needed to updated/created if child objects should be updated/created as well (if any)
- **wait_for_children** – when true child objects will be updated/created blocking (if any), else async
- **enabled_sections** – Sections to load regardless of current settings

Returns Queryset with all requested eve objects

get_or_create_esi(***, *id: int*, *include_children: bool = False*, *wait_for_children: bool = True*, *enabled_sections: Optional[Iterable[str]] = None*) → Tuple[django.db.models.base.Model, bool]

gets or creates an eve universe object.

The object is automatically fetched from ESI if it does not exist (blocking). Will always get/create parent objects.

Parameters

- **id** – Eve Online ID of object
- **include_children** – if child objects should be updated/created as well (only when a new object is created)
- **wait_for_children** – when true child objects will be updated/created blocking (if any), else async (only when a new object is created)
- **enabled_sections** – Sections to load regardless of current settings, e.g. *[EveType.Section.DOGMAS]* will always load dogmas for EveTypes

Returns A tuple consisting of the requested object and a created flag

update_or_create_all_esi(*, *include_children: bool = False*, *wait_for_children: bool = True*, *enabled_sections: Optional[Iterable[str]] = None*) → None

updates or creates all objects of this class from ESI.

Loading all objects can take a long time. Use with care!

Parameters

- **include_children** – if child objects should be updated/created as well (if any)
- **wait_for_children** – when false all objects will be loaded async, else blocking
- **enabled_sections** – Sections to load regardless of current settings

update_or_create_esi(*, *id: int*, *include_children: bool = False*, *wait_for_children: bool = True*, *enabled_sections: Optional[Iterable[str]] = None*) → Tuple[django.db.models.base.Model, bool]

updates or creates an Eve universe object by fetching it from ESI (blocking). Will always get/create parent objects

Parameters

- **id** – Eve Online ID of object
- **include_children** – if child objects should be updated/created as well (if any)
- **wait_for_children** – when true child objects will be updated/created blocking (if any), else async
- **enabled_sections** – Sections to load regardless of current settings, e.g. *[EveType.Section.DOGMAS]* will always load dogmas for EveTypes

Returns A tuple consisting of the requested object and a created flag

3.4.2 EveEntity manager methods

EveEntity comes with some additional manager methods.

class EveEntityQuerySet(*model=None*, *query=None*, *using=None*, *hints=None*)

Custom queryset for EveEntity

update_from_esi() → int

Updates all Eve entity objects in this queryset from ESI

class EveEntityManager(**args*, ***kwargs*)

Custom manager for EveEntity

bulk_create_esi(*ids: Iterable[int]*) → int

bulk create and resolve multiple entities from ESI. Will also resolve existing entities, that have no name.

Parameters **ids** – List of valid EveEntity IDs

Returns Count of updated entities

bulk_resolve_names(*ids: Iterable[int]*) → *eveuniverse.helpers.EveEntityNameResolver*
returns a map of IDs to names in a resolver object for given IDs

Parameters **ids** – List of valid EveEntity IDs

Returns EveEntityNameResolver object helpful for quick resolving a large amount of IDs

bulk_update_all_esi()
Updates all EveEntity objects in the database from ESI.

Returns Count of updated entities.

bulk_update_new_esi() → int
updates all unresolved EveEntity objects in the database from ESI.

Returns Count of updated entities.

get_or_create_esi(*, *id: int*, *include_children: bool = False*, *wait_for_children: bool = True*) → Tuple[django.db.models.base.Model, bool]
gets or creates an EvEntity object.

The object is automatically fetched from ESI if it does not exist (blocking) or if it has not yet been resolved.

Parameters **id** – Eve Online ID of object

Returns A tuple consisting of the requested EveEntity object and a created flag Returns a None objects if the ID is invalid

resolve_name(*id: int*) → str
return the name for the given Eve entity ID or an empty string if ID is not valid

update_or_create_esi(*, *id: int*, *include_children: bool = False*, *wait_for_children: bool = True*, *enabled_sections: Optional[Iterable[str]] = None*) → Tuple[Optional[django.db.models.base.Model], bool]
updates or creates an EveEntity object by fetching it from ESI (blocking).

Parameters

- **id** – Eve Online ID of object
- **include_children** – (no effect)
- **wait_for_children** – (no effect)

Returns A tuple consisting of the requested object and a created flag When the ID is invalid the returned object will be None

Exceptions: Raises all HTTP codes of ESI endpoint /universe/names except 404

3.4.3 Other manager methods

class EveMarketPriceManager(**args*, ***kwargs*)

update_from_esi(*minutes_until_stale: Optional[int] = None*) → int
Updates market prices from ESI. Will only create new price objects for EveTypes that already exist in the database.

Parameters **minutes_until_stale** – only prices older then given minutes are regarding as stale and will be updated. Will use default (60) if not specified.

Returns Count of updated types

3.5 Helpers

class EveEntityNameResolver(*names_map: Dict[int, str]*)
Container with a mapping between entity Ids and entity names and a performant API

to_name(*id: int*) → str
Resolved an entity ID to a name

Parameters **id** – ID of the Eve entity to resolve

Returns name for corresponding entity ID if known else an empty string

meters_to_au(*value: float*) → float
converts meters into AU

meters_to_ly(*value: float*) → float
converts meters into lightyears

3.6 Tasks

3.6.1 Eve Universe tasks

load_eve_object(*model_name: str*, *id: int*, *include_children=False*, *wait_for_children=True*) → None
Task for loading an eve object. Will only be created from ESI if it does not exist

update_or_create_eve_object(*model_name: str*, *id: int*, *include_children=False*, *wait_for_children=True*, *enabled_sections: List[str] = None*) → None
Task for updating or creating an eve object from ESI

3.6.2 EveEntity tasks

create_eve_entities(*ids: Iterable[int]*) → None
Task for bulk creating and resolving multiple entities from ESI.

update_unresolved_eve_entities() → None
Task for bulk updating all unresolved EveEntity objects in the database from ESI.

3.6.3 Object loader tasks

create_eve_entities(*ids: Iterable[int]*) → None
Task for bulk creating and resolving multiple entities from ESI.

update_unresolved_eve_entities() → None
Task for bulk updating all unresolved EveEntity objects in the database from ESI.

load_map() → None
loads the complete Eve map with all regions, constellation and solarsystems and additional related entities if they are enabled

load_eve_types(*category_ids: List[int] = None*, *group_ids: List[int] = None*, *type_ids: List[int] = None*, *force_loading_dogma: bool = False*) → None

Load specified eve types from ESI. Will always load all children except for EveType

Args: - category_ids: EveCategory IDs - group_ids: EveGroup IDs - type_ids: EveType IDs - load_dogma: When True will load dogma for all types

3.6.4 Other tasks

update_market_prices(*minutes_until_stale: int = None*)

Updates market prices from ESI. see EveMarketPrice.objects.update_from_esi() for details

3.7 Tools

3.7.1 Testdata

ModelSpec(*model_name: str*, *ids: List[int]*, *include_children: bool = False*, *enabled_sections: Optional[Iterable[str]] = None*) → *eveuniverse.tools.testdata.ModelSpec*

Wrapper for creating ModelSpec objects.

A Modelspec class defines what objects are to be loaded as test data

Parameters

- **model_name** – Name of Eve Universe model
- **ids** – List of Eve IDs to be loaded
- **include_children** – Whether to also load children of those objects

create_testdata(*spec: List[*eveuniverse.tools.testdata.ModelSpec*]*, *filepath: str*) → None

Loads eve data from ESI as defined by spec and dumps it to file as JSON

Parameters

- **spec** – Specification of which Eve objects to load. The specification can contain the same model more than once.
- **filepath** – absolute path of where to store the resulting JSON file

load_testdata_from_dict(*testdata: dict*) → None

creates eve objects in the database from testdata dump given as dict

Parameters **testdata** – The dict containing the testdata as created by *create_testdata()*

load_testdata_from_file(*filepath: str*) → None

creates eve objects in the database from testdata dump given as JSON file

Parameters **filepath** – Absolute path to the JSON file containing the testdata created by *create_testdata()*

See also:

Please also see *Test data* on how to create test data for your app.

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

e

INDEX

Symbols

A

B

C

D

E

F

G

I

J

L

M

N

P

R

S

T

U