
Highton Documentation

Release 2.0.0

Michael Bykovski, Julia Giebelhausen, Jakob Löhnertz

Jul 30, 2018

1	Quickstart	1
1.1	Install	1
1.2	Setup	1
1.3	Retrieve data	1
1.4	Get a single object	1
1.5	Create data	2
1.6	Update data	2
1.7	Delete data	2
2	Models	3
2.1	Address	3
2.2	AssociatedParty	4
2.3	Attachment	5
2.4	Case	6
2.5	Category	9
2.6	Comment	10
2.7	Company	11
2.8	Contact	15
2.9	ContactData	16
2.10	Deal	17
2.11	DealCategory	21
2.12	Email	22
2.13	EmailAddress	24
2.14	Group	25
2.15	HightonModel	27
2.16	InstantMessenger	27
2.17	Note	28
2.18	Party	30
2.19	Person	32
2.20	PhoneNumber	35
2.21	SubjectData	36
2.22	Tag	37
2.23	Task	38
2.24	TaskCategory	40
2.25	User	42
2.26	TwitterAccount	44

2.27	WebAddress	44
3	Fields	47
3.1	BooleanField	47
3.2	DateField	48
3.3	DateTimeField	48
3.4	Field	49
3.5	IntegerField	49
3.6	ListField	50
3.7	ObjectField	50
3.8	StringField	51
4	Indices and tables	53
	Python Module Index	55

1.1 Install

```
pip install highton
```

1.2 Setup

Just setup highton by defining a singleton in your application. You don't have to store this singleton in a variable.

```
from highton.highton_settings import HightonSettings  
  
HightonSettings(username='<your_username>', api_key='<your_API_KEY>')
```

1.3 Retrieve data

If you have defined the 'USERNAME' and 'API_KEY', you can start to retrieve data:

```
from highton.models import Deal  
  
deals = Deal.list()
```

1.4 Get a single object

```
from highton.models import Deal  
  
deal_object = Deal.get(12345)
```

1.5 Create data

```
from highton.models import Deal

new_deal_object = Deal(name='new deal').create()
```

1.6 Update data

```
from highton.models import Deal

deal_object = Deal.get(12345)
deal_object.name = 'new name'
deal_object.update()
```

1.7 Delete data

```
from highton.models import Deal

deal_object = Deal.get(12345)
deal_object.delete()
```

2.1 Address

```
class highton.models.address.Address (**kwargs)
```

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **location** – fields.StringField(name=HightonConstants.LOCATION)
- **city** – fields.StringField(name=HightonConstants.CITY)
- **country** – fields.StringField(name=HightonConstants.COUNTRY)
- **state** – fields.StringField(name=HightonConstants.STATE)
- **street** – fields.StringField(name=HightonConstants.STREET)
- **zip** – fields.StringField(name=HightonConstants.ZIP)

```
ENDPOINT = None
```

```
TAG_NAME = 'address'
```

```
classmethod decode (root_element)
```

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

```
static element_from_string (string)
```

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (element)
Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()
Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

to_serializable_value ()
Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

2.2 AssociatedParty

```
class highton.models.associated_party.AssociatedParty (**kwargs)
```

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **author_id** – fields.IntegerField(name=HightonConstants.AUTHOR_ID)
- **background** – fields.StringField(name=HightonConstants.BACKGROUND)
- **company_id** – fields.IntegerField(name=HightonConstants.COMPANY_ID)
- **created_at** – fields.DatetimeField(name=HightonConstants.CREATED_AT)
- **first_name** – fields.StringField(name=HightonConstants.FIRST_NAME)
- **name** – fields.StringField(name=HightonConstants.NAME)
- **group_id** – fields.IntegerField(name=HightonConstants.GROUP_ID)
- **last_name** – fields.StringField(name=HightonConstants.LAST_NAME)
- **owner_id** – fields.IntegerField(name=HightonConstants.OWNER_ID)
- **title** – fields.StringField(name=HightonConstants.TITLE)
- **updated_at** – fields.DatetimeField(name=HightonConstants.UPDATED_AT)
- **visible_to** – fields.StringField(name=HightonConstants.VISIBLE_TO)
- **company_name** – fields.StringField(name=HightonConstants.COMPANY_NAME)
- **linkedin_url** – fields.StringField(name=HightonConstants.LINKEDIN_URL)
- **avatar_url** – fields.StringField(name=HightonConstants.AVATAR_URL)
- **type** – fields.StringField(name=HightonConstants.TYPE)
- **tags** – fields.ListField(name=HightonConstants.TAGS, init_class=Tag)

- **contact_data** – fields.ObjectField(name=HightonConstants.CONTACT_DATA, init_class=ContactData)
- **subject_datas** – fields.ListField(name=HightonConstants.SUBJECT_DATAS, init_class=SubjectData)

ENDPOINT = 'parties'

TAG_NAME = 'associated-party'

classmethod decode (*root_element*)
Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

static element_from_string (*string*)
Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)
Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()
Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

to_serializable_value ()
Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

2.3 Attachment

class highton.models.attachment.**Attachment** (***kwargs*)

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **url** – fields.StringField(name=HightonConstants.URL)
- **name** – fields.StringField(name=HightonConstants.NAME)

- **size** – fields.IntegerField(name=HightonConstants.SIZE)

```
COLLECTION_DATETIME = '%Y%m%d%H%M%S'
```

```
ENDPOINT = None
```

```
TAG_NAME = 'attachment'
```

```
classmethod decode (root_element)
```

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

```
static element_from_string (string)
```

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

```
static element_to_string (element)
```

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

```
encode ()
```

Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

```
get ()
```

```
to_serializable_value ()
```

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

2.4 Case

```
class highton.models.case.Case (**kwargs)
```

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **author_id** – fields.IntegerField(name=HightonConstants.AUTHOR_ID)
- **closed_at** – fields.DatetimeField(name=HightonConstants.CLOSED_AT)
- **created_at** – fields.DatetimeField(name=HightonConstants.CREATED_AT)

- **updated_at** – fields.DatetimeField(name=HightonConstants.UPDATED_AT)
- **name** – fields.StringField(name=HightonConstants.NAME)
- **visible_to** – fields.StringField(name=HightonConstants.VISIBLE_TO)
- **group_id** – fields.IntegerField(name=HightonConstants.GROUP_ID)
- **owner_id** – fields.IntegerField(name=HightonConstants.OWNER_ID)
- **background** – fields.StringField(name=HightonConstants.BACKGROUND)
- **parties** – fields.ListField(name=HightonConstants.PARTIES, init_class=Party)
- **associated_parties** – fields.ListField(name=HightonConstants.ASSOCIATED_PARTIES, init_class=AssociatedParty)

COLLECTION_DATETIME = '%Y%m%d%H%M%S'

EMAILS_OFFSET = 25

ENDPOINT = 'kases'

NOTES_OFFSET = 25

TAG_NAME = 'kase'

add_note (*body*)

Create a Note to current object

Parameters **body** (*str*) – the body of the note

Returns newly created Note

Return type *Tag*

add_tag (*name*)

Create a Tag to current object

Parameters **name** (*str*) – the name of the tag

Returns newly created Tag

Return type *Tag*

create ()

Creates the object

Returns the created object

Return type object

classmethod decode (*root_element*)

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

delete ()

Deletes the object

Returns

Return type None

delete_tag (*tag_id*)

Deletes a Tag to current object

Parameters **tag_id** (*int*) – the id of the tag which should be deleted

Return type None

static element_from_string (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()

Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

classmethod get (*object_id*)

Retrieves a single model

Parameters **object_id** (*integer*) – the primary id of the model

Returns the object of the parsed xml object

Return type object

classmethod list (*params=None*)

Retrieves a list of the model

Parameters **params** (*dict*) – params as dictionary

Returns the list of the parsed xml objects

Return type list

list_emails (*page=0, since=None*)

Get the emails of current object

Parameters

- **page** – the page starting at 0
- **since** (*datetime.datetime*) – get all notes since a datetime

Returns the emails

Return type list

list_notes (*page=0, since=None*)

Get the notes of current object

Parameters

- **page** – the page starting at 0

- **since** (*datetime.datetime*) – get all notes since a datetime

Returns the notes

Return type list

list_tags ()

Get the tags of current object

Returns the tags

Return type list

list_tasks ()

Get the tasks of current object

Returns the tasks

Return type list

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update ()

Updates the object

Returns

Return type response

2.5 Category

```
class highton.models.category.Category (**kwargs)
```

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **name** – fields.StringField(name=HightonConstants.NAME)

ENDPOINT = None

TAG_NAME = 'category'

classmethod decode (*root_element*)

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

static element_from_string (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type `xml.etree.ElementTree.Element`

static element_to_string (*element*)
Parses the `xml.etree.ElementTree.Element` to a string

Parameters *element* (`xml.etree.ElementTree.Element`) – a xml element

Returns the parsed string

Return type `str`

encode ()
Encodes the object to a `xml.etree.ElementTree.Element`

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

to_serializable_value ()
Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type `dict`

2.6 Comment

```
class highton.models.comment.Comment (**kwargs)
```

Variables

- **id** – `fields.IntegerField(name=HightonConstants.ID)`
- **parent_id** – `fields.IntegerField(name=HightonConstants.PARENT_ID)`
- **author_id** – `fields.IntegerField(name=HightonConstants.AUTHOR_ID)`
- **created_at** – `fields.DateTimeField(name=HightonConstants.CREATED_AT)`
- **body** – `fields.StringField(name=HightonConstants.BODY)`

```
COLLECTION_DATETIME = '%Y%m%d%H%M%S'
```

```
ENDPOINT = 'comments'
```

```
TAG_NAME = 'comment'
```

```
create ()
```

Creates the object

Returns the created object

Return type `object`

```
classmethod decode (root_element)
```

Decode the object to the object

Parameters *root_element* (`xml.etree.ElementTree.Element`) – the parsed xml Element

Returns the decoded Element as object

Return type `object`

delete()
 Deletes the object

Returns

Return type None

static element_from_string(string)
 Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string(element)
 Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode()
 Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

classmethod get(object_id)
 Retrieves a single model

Parameters **object_id** (*integer*) – the primary id of the model

Returns the object of the parsed xml object

Return type object

to_serializable_value()
 Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update()
 Updates the object

Returns

Return type response

2.7 Company

```
class highton.models.company.Company(**kwargs)
```

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **author_id** – fields.IntegerField(name=HightonConstants.AUTHOR_ID)

- **background** – fields.StringField(name=HightonConstants.BACKGROUND)
- **created_at** – fields.DateTimeField(name=HightonConstants.CREATED_AT)
- **group_id** – fields.IntegerField(name=HightonConstants.GROUP_ID)
- **owner_id** – fields.IntegerField(name=HightonConstants.OWNER_ID)
- **updated_at** – fields.DateTimeField(name=HightonConstants.UPDATED_AT)
- **visible_to** – fields.StringField(name=HightonConstants.VISIBLE_TO)
- **linkedin_url** – fields.StringField(name=HightonConstants.LINKEDIN_URL)
- **avatar_url** – fields.StringField(name=HightonConstants.AVATAR_URL)
- **tags** – fields.ListField(name=HightonConstants.TAGS, init_class=Tag)
- **contact_data** – fields.ObjectField(name=HightonConstants.CONTACT_DATA, init_class=ContactData)
- **subject_datas** – fields.ListField(name=HightonConstants.SUBJECT_DATAS, init_class=SubjectData)
- **name** – fields.StringField(name=HightonConstants.NAME)

COLLECTION_DATETIME = '%Y%m%d%H%M%S'

EMAILS_OFFSET = 25

ENDPOINT = 'companies'

NOTES_OFFSET = 25

OFFSET = 500

SEARCH_OFFSET = 25

TAG_NAME = 'company'

add_note (*body*)

Create a Note to current object

Parameters **body** (*str*) – the body of the note

Returns newly created Note

Return type *Tag*

add_tag (*name*)

Create a Tag to current object

Parameters **name** (*str*) – the name of the tag

Returns newly created Tag

Return type *Tag*

create ()

Creates the object

Returns the created object

Return type object

classmethod decode (*root_element*)

Decode the object to the object

Parameters `root_element` (`xml.etree.ElementTree.Element`) – the parsed xml Element

Returns the decoded Element as object

Return type object

delete()
Deletes the object

Returns

Return type None

delete_tag(tag_id)
Deletes a Tag to current object

Parameters `tag_id` (`int`) – the id of the tag which should be deleted

Return type None

static element_from_string(string)
Make a Element from a str

Parameters `string` (`str`) – string you want to parse

Returns the parsed xml string

Return type `xml.etree.ElementTree.Element`

static element_to_string(element)
Parses the `xml.etree.ElementTree.Element` to a string

Parameters `element` (`xml.etree.ElementTree.Element`) – a xml element

Returns the parsed string

Return type str

encode()
Encodes the object to a `xml.etree.ElementTree.Element`

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

classmethod get(object_id)
Retrieves a single model

Parameters `object_id` (`integer`) – the primary id of the model

Returns the object of the parsed xml object

Return type object

classmethod list(page=0, since=None, tag_id=None, title=None)

Parameters

- **page** (`int`) – page starting by 0
- **since** (`datetime.datetime`) –
- **tag_id** (`int`) – id of a tag
- **title** (`str`) –

Returns list of person objects

Return type list

list_emails (*page=0, since=None*)

Get the emails of current object

Parameters

- **page** – the page starting at 0
- **since** (*datetime.datetime*) – get all notes since a datetime

Returns the emails

Return type list

list_notes (*page=0, since=None*)

Get the notes of current object

Parameters

- **page** – the page starting at 0
- **since** (*datetime.datetime*) – get all notes since a datetime

Returns the notes

Return type list

list_tags ()

Get the tags of current object

Returns the tags

Return type list

list_tasks ()

Get the tasks of current object

Returns the tasks

Return type list

classmethod search (*term=None, page=0, **criteria*)

Search a list of the model If you use “term”: - Returns a collection of people that have a name matching the term passed in through the URL.

If you use “criteria”: - returns people who match your search criteria. Search by any criteria you can on the Contacts tab, including custom fields. Combine criteria to narrow results

Parameters

- **term** (*str*) – params as string
- **criteria** (*dict*) – search for more criteria
- **page** (*int*) – the page

Returns the list of the parsed xml objects

Return type list

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update()

Updates the object

Returns

Return type response

2.8 Contact

class `highton.models.contact.Contact` (***kwargs*)

Variables

- **id** – `fields.IntegerField(name=HightonConstants.ID)`
- **author_id** – `fields.IntegerField(name=HightonConstants.AUTHOR_ID)`
- **background** – `fields.StringField(name=HightonConstants.BACKGROUND)`
- **created_at** – `fields.DateTimeField(name=HightonConstants.CREATED_AT)`
- **group_id** – `fields.IntegerField(name=HightonConstants.GROUP_ID)`
- **owner_id** – `fields.IntegerField(name=HightonConstants.OWNER_ID)`
- **updated_at** – `fields.DateTimeField(name=HightonConstants.UPDATED_AT)`
- **visible_to** – `fields.StringField(name=HightonConstants.VISIBLE_TO)`
- **linkedin_url** – `fields.StringField(name=HightonConstants.LINKEDIN_URL)`
- **avatar_url** – `fields.StringField(name=HightonConstants.AVATAR_URL)`
- **tags** – `fields.ListField(name=HightonConstants.TAGS, init_class=Tag)`
- **contact_data** – `fields.ObjectField(name=HightonConstants.CONTACT_DATA, init_class=ContactData)`
- **subject_datas** – `fields.ListField(name=HightonConstants.SUBJECT_DATAS, init_class=SubjectData)`

ENDPOINT = `None`

TAG_NAME = `None`

classmethod `decode` (*root_element*)

Decode the object to the object

Parameters **root_element** (`xml.etree.ElementTree.Element`) – the parsed xml Element

Returns the decoded Element as object

Return type object

static `element_from_string` (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type `xml.etree.ElementTree.Element`

static `element_to_string` (*element*)

Parses the `xml.etree.ElementTree.Element` to a string

Parameters `element` (`xml.etree.ElementTree.Element`) – a xml element

Returns the parsed string

Return type `str`

encode()

Encodes the object to a `xml.etree.ElementTree.Element`

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

to_serializable_value()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type `dict`

2.9 ContactData

```
class highton.models.contact_data.ContactData (**kwargs)
```

Variables

- **id** – `fields.IntegerField(name=HightonConstants.ID)`
- **phone_numbers** – `fields.ListField(name=HightonConstants.PHONE_NUMBERS, init_class=PhoneNumber)`
- **twitter_accounts** – `fields.ListField(name=HightonConstants.TWITTER_ACCOUNTS, init_class=TwitterAccount)`
- **web_addresses** – `fields.ListField(name=HightonConstants.WEB_ADDRESSES, init_class=WebAddress)`
- **email_addresses** – `fields.ListField(name=HightonConstants.EMAIL_ADDRESSES, init_class=EmailAddress)`
- **addresses** – `fields.ListField(name=HightonConstants.ADDRESSES, init_class=Address)`
- **instant_messenger** – `fields.ListField(name=HightonConstants.INSTANT_MESSENGERS, init_class=InstantMessenger)`

ENDPOINT = `None`

TAG_NAME = `'contact-data'`

classmethod decode (`root_element`)

Decode the object to the object

Parameters `root_element` (`xml.etree.ElementTree.Element`) – the parsed xml Element

Returns the decoded Element as object

Return type `object`

static element_from_string (`string`)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()

Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

2.10 Deal

class highton.models.deal.Deal (***kwargs*)

A deal which represents:

<https://github.com/basecamp/highrise-api/blob/master/sections/deals.md>

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **author_id** – fields.IntegerField(name=HightonConstants.AUTHOR_ID)
- **account_id** – fields.IntegerField(name=HightonConstants.ACCOUNT_ID)
- **background** – fields.StringField(name=HightonConstants.BACKGROUND)
- **category_id** – fields.IntegerField(name=HightonConstants.CATEGORY_ID)
- **created_at** – fields.DateTimeField(name=HightonConstants.CREATED_AT)
- **currency** – fields.StringField(name=HightonConstants.CURRENCY)
- **duration** – fields.IntegerField(name=HightonConstants.DURATION)
- **group_id** – fields.IntegerField(name=HightonConstants.GROUP_ID)
- **name** – fields.StringField(name=HightonConstants.NAME)
- **owner_id** – fields.IntegerField(name=HightonConstants.OWNER_ID)
- **party_id** – fields.IntegerField(name=HightonConstants.PARTY_ID)
- **price** – fields.IntegerField(name=HightonConstants.PRICE)
- **price_type** – fields.StringField(name=HightonConstants.PRICE_TYPE)

- **responsible_party_id** – fields.IntegerField(name=HightonConstants.RESPONSIBLE_PARTY_ID)
- **status** – fields.StringField(name=HightonConstants.STATUS)
- **status_changed_on** – fields.DateField(name=HightonConstants.STATUS_CHANGED_ON)
- **updated_at** – fields.DateTimeField(name=HightonConstants.UPDATED_AT)
- **visible_to** – fields.StringField(name=HightonConstants.VISIBLE_TO)
- **party** – fields.ObjectField(name=HightonConstants.PARTY, init_class=Party)
- **category** – fields.ObjectField(name=HightonConstants.CATEGORY, init_class=Category)
- **tags** – fields.ListField(name=HightonConstants.TAGS, init_class=Tag)
- **parties** – fields.ListField(name=HightonConstants.PARTIES, init_class=Party)
- **contact_data** – fields.ObjectField(name=HightonConstants.CONTACT_DATA, init_class=ContactData)
- **subject_datas** – fields.ListField(name=HightonConstants.SUBJECT_DATAS, init_class=SubjectData)
- **associated_parties** – fields.ListField(name=HightonConstants.ASSOCIATED_PARTIES, init_class=AssociatedParty)

COLLECTION_DATETIME = '%Y%m%d%H%M%S'

EMAILS_OFFSET = 25

ENDPOINT = 'deals'

NOTES_OFFSET = 25

OFFSET = 500

TAG_NAME = 'deal'

add_note (*body*)

Create a Note to current object

Parameters **body** (*str*) – the body of the note

Returns newly created Note

Return type *Tag*

add_tag (*name*)

Create a Tag to current object

Parameters **name** (*str*) – the name of the tag

Returns newly created Tag

Return type *Tag*

create ()

Creates the object

Returns the created object

Return type object

classmethod decode (*root_element*)

Decode the object to the object

Parameters `root_element` (`xml.etree.ElementTree.Element`) – the parsed xml Element

Returns the decoded Element as object

Return type object

delete()

Deletes the object

Returns

Return type None

delete_tag (`tag_id`)

Deletes a Tag to current object

Parameters `tag_id` (`int`) – the id of the tag which should be deleted

Return type None

static element_from_string (`string`)

Make a Element from a str

Parameters `string` (`str`) – string you want to parse

Returns the parsed xml string

Return type `xml.etree.ElementTree.Element`

static element_to_string (`element`)

Parses the `xml.etree.ElementTree.Element` to a string

Parameters `element` (`xml.etree.ElementTree.Element`) – a xml element

Returns the parsed string

Return type str

encode()

Encodes the object to a `xml.etree.ElementTree.Element`

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

classmethod get (`object_id`)

Retrieves a single model

Parameters `object_id` (`integer`) – the primary id of the model

Returns the object of the parsed xml object

Return type object

classmethod list (`page=0, status=None, since=None`)

Parameters

- **page** (`int`) – page starting by 0
- **since** (`datetime.datetime`) –
- **status** (`str`) –

Returns list of person objects

Return type list

list_emails (*page=0, since=None*)

Get the emails of current object

Parameters

- **page** – the page starting at 0
- **since** (*datetime.datetime*) – get all notes since a datetime

Returns the emails

Return type list

list_notes (*page=0, since=None*)

Get the notes of current object

Parameters

- **page** – the page starting at 0
- **since** (*datetime.datetime*) – get all notes since a datetime

Returns the notes

Return type list

list_tags ()

Get the tags of current object

Returns the tags

Return type list

list_tasks ()

Get the tasks of current object

Returns the tasks

Return type list

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update ()

Updates the object

Returns

Return type response

update_status (*status*)

Updates the status of the deal

Parameters **status** – status have to be ('won', 'pending', 'lost')

Returns successfull response or raise Exception

Return type

2.11 DealCategory

```
class highton.models.deal_category.DealCategory (**kwargs)
```

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **name** – fields.StringField(name=HightonConstants.NAME)
- **color** – fields.StringField(name=HightonConstants.COLOR)
- **account_id** – fields.IntegerField(name=HightonConstants.ACCOUNT_ID)
- **created_at** – fields.DateTimeField(name=HightonConstants.CREATED_AT)
- **updated_at** – fields.DateTimeField(name=HightonConstants.UPDATED_AT)
- **elements_count** – fields.IntegerField(name=HightonConstants.ELEMENTS_COUNT)

```
COLLECTION_DATETIME = '%Y%m%d%H%M%S'
```

```
ENDPOINT = 'deal_categories'
```

```
TAG_NAME = 'deal-category'
```

```
create ()
```

Creates the object

Returns the created object

Return type object

```
classmethod decode (root_element)
```

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

```
delete ()
```

Deletes the object

Returns

Return type None

```
static element_from_string (string)
```

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

```
static element_to_string (element)
```

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode()
Encodes the object to a `xml.etree.ElementTree.Element`

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

classmethod get(object_id)
Retrieves a single model

Parameters **object_id** (*integer*) – the primary id of the model

Returns the object of the parsed xml object

Return type object

classmethod list(params=None)
Retrieves a list of the model

Parameters **params** (*dict*) – params as dictionary

Returns the list of the parsed xml objects

Return type list

to_serializable_value()
Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update()
Updates the object

Returns

Return type response

2.12 Email

```
class highton.models.email.Email(**kwargs)
```

Variables

- **id** – `fields.IntegerField(name=HightonConstants.ID)`
- **title** – `fields.StringField(name=HightonConstants.TITLE, required=True)`
- **body** – `fields.StringField(name=HightonConstants.BODY, required=True)`
- **subject_id** – `fields.IntegerField(name=HightonConstants.SUBJECT_ID, required=True)`
- **subject_type** – `fields.StringField(name=HightonConstants.SUBJECT_TYPE, required=True)`
- **subject_name** – `fields.StringField(name=HightonConstants.SUBJECT_NAME)`
- **author_id** – `fields.IntegerField(name=HightonConstants.AUTHOR_ID)`
- **collection_id** – `fields.IntegerField(name=HightonConstants.COLLECTION_ID)`
- **collection_type** – `fields.StringField(name=HightonConstants.COLLECTION_TYPE)`

- **visible_to** – fields.StringField(name=HightonConstants.VISIBLE_TO)
- **owner_id** – fields.IntegerField(name=HightonConstants.OWNER_ID)
- **group_id** – fields.IntegerField(name=HightonConstants.GROUP_ID)
- **updated_at** – fields.DatetimeField(name=HightonConstants.UPDATED_AT)
- **created_at** – fields.DatetimeField(name=HightonConstants.CREATED_AT)
- **attachments** – fields.ListField(name=HightonConstants.ATTACHMENTS, init_class=Attachment)

COLLECTION_DATETIME = '%Y%m%d%H%M%S'

COMMENT_OFFSET = 25

ENDPOINT = 'emails'

TAG_NAME = 'email'

create()

Creates the object

Returns the created object

Return type object

classmethod decode(*root_element*)

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

delete()

Deletes the object

Returns

Return type None

static element_from_string(*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string(*element*)

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode()

Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

classmethod `get(object_id)`

Retrieves a single model

Parameters `object_id` (*integer*) – the primary id of the model

Returns the object of the parsed xml object

Return type object

list_comments (*page=0*)

Get the comments of current object

Parameters `page` – the page starting at 0

Returns the emails

Return type list

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update ()

Updates the object

Returns

Return type response

2.13 EmailAddress

class `highton.models.email_address.EmailAddress(**kwargs)`

Variables

- `id` – `fields.IntegerField(name=HightonConstants.ID)`
- `location` – `fields.StringField(name=HightonConstants.LOCATION)`
- `address` – `fields.StringField(name=HightonConstants.ADDRESS)`

ENDPOINT = `None`

TAG_NAME = `'email-address'`

classmethod `decode(root_element)`

Decode the object to the object

Parameters `root_element` (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

static `element_from_string(string)`

Make a Element from a str

Parameters `string` (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)
 Parses the xml.etree.ElementTree.Element to a string

Parameters *element* (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()
 Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

to_serializable_value ()
 Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

2.14 Group

class highton.models.group.**Group** (**kwargs)
 The authenticated user needs to be an administrator to perform these actions.

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **name** – fields.StringField(name=HightonConstants.NAME)
- **users** – fields.ListField(name=HightonConstants.USERS, init_class=models.User)

COLLECTION_DATETIME = '%Y%m%d%H%M%S'

ENDPOINT = 'groups'

TAG_NAME = 'group'

create ()
 Creates the object

Returns the created object

Return type object

classmethod decode (*root_element*)
 Decode the object to the object

Parameters *root_element* (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

delete ()
 Deletes the object

Returns

Return type None

static element_from_string (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()

Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

classmethod get (*object_id*)

Retrieves a single model

Parameters **object_id** (*integer*) – the primary id of the model

Returns the object of the parsed xml object

Return type object

classmethod list (*params=None*)

Retrieves a list of the model

Parameters **params** (*dict*) – params as dictionary

Returns the list of the parsed xml objects

Return type list

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update ()

Updates the object

Returns

Return type response

2.15 HightonModel

class `highton.models.highton_model.HightonModel` (***kwargs*)

This class inherit from XMLDecoder and XMLEncoder which allows every HightonModel to be encoded and decoded in xml.

Variables `id` – `fields.IntegerField(name=HightonConstants.ID)`

ENDPOINT = `None`

TAG_NAME = `None`

classmethod `decode` (*root_element*)

Decode the object to the object

Parameters `root_element` (`xml.etree.ElementTree.Element`) – the parsed xml Element

Returns the decoded Element as object

Return type object

static `element_from_string` (*string*)

Make a Element from a str

Parameters `string` (*str*) – string you want to parse

Returns the parsed xml string

Return type `xml.etree.ElementTree.Element`

static `element_to_string` (*element*)

Parses the `xml.etree.ElementTree.Element` to a string

Parameters `element` (`xml.etree.ElementTree.Element`) – a xml element

Returns the parsed string

Return type `str`

encode ()

Encodes the object to a `xml.etree.ElementTree.Element`

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type `dict`

2.16 InstantMessenger

class `highton.models.instant_messenger.InstantMessenger` (***kwargs*)

Variables

- `id` – `fields.IntegerField(name=HightonConstants.ID)`
- `location` – `fields.StringField(name=HightonConstants.LOCATION)`

- **address** – fields.StringField(name=HightonConstants.ADDRESS)
- **protocol** – fields.StringField(name=HightonConstants.PROTOCOL)

ENDPOINT = None

TAG_NAME = 'instant-messenger'

classmethod decode (*root_element*)
Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

static element_from_string (*string*)
Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)
Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()
Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

to_serializable_value ()
Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

2.17 Note

```
class highton.models.note.Note (**kwargs)
```

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **body** – fields.StringField(name=HightonConstants.BODY, required=True)
- **author_id** – fields.IntegerField(name=HightonConstants.AUTHOR_ID)
- **subject_id** – fields.IntegerField(name=HightonConstants.SUBJECT_ID, required=True)

- **subject_type** – fields.StringField(name=HightonConstants.SUBJECT_TYPE, required=True)
- **subject_name** – fields.StringField(name=HightonConstants.SUBJECT_NAME)
- **collection_id** – fields.IntegerField(name=HightonConstants.COLLECTION_ID)
- **collection_type** – fields.StringField(name=HightonConstants.COLLECTION_TYPE)
- **visible_to** – fields.StringField(name=HightonConstants.VISIBLE_TO)
- **owner_id** – fields.IntegerField(name=HightonConstants.OWNER_ID)
- **group_id** – fields.IntegerField(name=HightonConstants.GROUP_ID)
- **updated_at** – fields.DateTimeField(name=HightonConstants.UPDATED_AT)
- **created_at** – fields.DateTimeField(name=HightonConstants.CREATED_AT)
- **attachments** – fields.ListField(name=HightonConstants.ATTACHMENTS, init_class=Attachment)

COLLECTION_DATETIME = '%Y%m%d%H%M%S'

COMMENT_OFFSET = 25

ENDPOINT = 'notes'

TAG_NAME = 'note'

create()

Creates the object

Returns the created object

Return type object

classmethod decode(*root_element*)

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

delete()

Deletes the object

Returns

Return type None

static element_from_string(*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string(*element*)

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode()

Encodes the object to a `xml.etree.ElementTree.Element`

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

classmethod get(object_id)

Retrieves a single model

Parameters **object_id** (*integer*) – the primary id of the model

Returns the object of the parsed xml object

Return type object

list_comments (*page=0*)

Get the comments of current object

Parameters **page** – the page starting at 0

Returns the emails

Return type list

to_serializable_value()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update()

Updates the object

Returns

Return type response

2.18 Party

class `highton.models.party.Party` (***kwargs*)

Variables

- **id** – `fields.IntegerField(name=HightonConstants.ID)`
- **author_id** – `fields.IntegerField(name=HightonConstants.AUTHOR_ID)`
- **background** – `fields.StringField(name=HightonConstants.BACKGROUND)`
- **company_id** – `fields.IntegerField(name=HightonConstants.COMPANY_ID)`
- **created_at** – `fields.DateTimeField(name=HightonConstants.CREATED_AT)`
- **first_name** – `fields.StringField(name=HightonConstants.FIRST_NAME)`
- **name** – `fields.StringField(name=HightonConstants.NAME)`
- **group_id** – `fields.IntegerField(name=HightonConstants.GROUP_ID)`
- **last_name** – `fields.StringField(name=HightonConstants.LAST_NAME)`

- **owner_id** – fields.IntegerField(name=HightonConstants.OWNER_ID)
- **title** – fields.StringField(name=HightonConstants.TITLE)
- **updated_at** – fields.DateTimeField(name=HightonConstants.UPDATED_AT)
- **visible_to** – fields.StringField(name=HightonConstants.VISIBLE_TO)
- **company_name** – fields.StringField(name=HightonConstants.COMPANY_NAME)
- **linkedin_url** – fields.StringField(name=HightonConstants.LINKEDIN_URL)
- **avatar_url** – fields.StringField(name=HightonConstants.AVATAR_URL)
- **type** – fields.StringField(name=HightonConstants.TYPE)
- **tags** – fields.ListField(name=HightonConstants.TAGS, init_class=Tag)
- **contact_data** – fields.ObjectField(name=HightonConstants.CONTACT_DATA, init_class=ContactData)
- **subject_datas** – fields.ListField(name=HightonConstants.SUBJECT_DATAS, init_class=SubjectData)

ENDPOINT = 'parties'

TAG_NAME = 'party'

classmethod decode (*root_element*)

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

static element_from_string (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()

Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

2.19 Person

```
class highton.models.person.Person (**kwargs)
```

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **company_id** – fields.IntegerField(name=HightonConstants.COMPANY_ID)
- **company_name** – fields.StringField(name=HightonConstants.COMPANY_NAME)
- **first_name** – fields.StringField(name=HightonConstants.FIRST_NAME)
- **last_name** – fields.StringField(name=HightonConstants.LAST_NAME)
- **title** – fields.StringField(name=HightonConstants.TITLE)
- **author_id** – fields.IntegerField(name=HightonConstants.AUTHOR_ID)
- **background** – fields.StringField(name=HightonConstants.BACKGROUND)
- **created_at** – fields.DateTimeField(name=HightonConstants.CREATED_AT)
- **group_id** – fields.IntegerField(name=HightonConstants.GROUP_ID)
- **owner_id** – fields.IntegerField(name=HightonConstants.OWNER_ID)
- **updated_at** – fields.DateTimeField(name=HightonConstants.UPDATED_AT)
- **visible_to** – fields.StringField(name=HightonConstants.VISIBLE_TO)
- **linkedin_url** – fields.StringField(name=HightonConstants.LINKEDIN_URL)
- **avatar_url** – fields.StringField(name=HightonConstants.AVATAR_URL)
- **tags** – fields.ListField(name=HightonConstants.TAGS, init_class=Tag)
- **contact_data** – fields.ObjectField(name=HightonConstants.CONTACT_DATA, init_class=ContactData)
- **subject_datas** – fields.ListField(name=HightonConstants.SUBJECT_DATAS, init_class=SubjectData)

```
COLLECTION_DATETIME = '%Y%m%d%H%M%S'
```

```
EMAILS_OFFSET = 25
```

```
ENDPOINT = 'people'
```

```
NOTES_OFFSET = 25
```

```
OFFSET = 500
```

```
SEARCH_OFFSET = 25
```

```
TAG_NAME = 'person'
```

```
add_note(body)
```

Create a Note to current object

Parameters **body** (*str*) – the body of the note

Returns newly created Note

Return type *Tag*

add_tag (*name*)

Create a Tag to current object

Parameters **name** (*str*) – the name of the tag

Returns newly created Tag

Return type *Tag*

create ()

Creates the object

Returns the created object

Return type object

classmethod decode (*root_element*)

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

delete ()

Deletes the object

Returns

Return type None

delete_tag (*tag_id*)

Deletes a Tag to current object

Parameters **tag_id** (*int*) – the id of the tag which should be deleted

Return type None

static element_from_string (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()

Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

classmethod `get (object_id)`

Retrieves a single model

Parameters `object_id (integer)` – the primary id of the model

Returns the object of the parsed xml object

Return type object

classmethod `list (page=0, since=None, tag_id=None, title=None)`

Parameters

- **page** (*int*) – page starting by 0
- **since** (*datetime.datetime*) –
- **tag_id** (*int*) – id of a tag
- **title** (*str*) –

Returns list of person objects

Return type list

list_emails (*page=0, since=None*)

Get the emails of current object

Parameters

- **page** – the page starting at 0
- **since** (*datetime.datetime*) – get all notes since a datetime

Returns the emails

Return type list

list_notes (*page=0, since=None*)

Get the notes of current object

Parameters

- **page** – the page starting at 0
- **since** (*datetime.datetime*) – get all notes since a datetime

Returns the notes

Return type list

list_tags ()

Get the tags of current object

Returns the tags

Return type list

list_tasks ()

Get the tasks of current object

Returns the tasks

Return type list

classmethod `search (term=None, page=0, **criteria)`

Search a list of the model If you use “term”: - Returns a collection of people that have a name matching the term passed in through the URL.

If you use “criteria”: - returns people who match your search criteria. Search by any criteria you can on the Contacts tab, including custom fields. Combine criteria to narrow results

Parameters

- **term** (*str*) – params as string
- **criteria** (*dict*) – search for more criteria
- **page** (*int*) – the page

Returns the list of the parsed xml objects

Return type list

to_serializable_value()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update()

Updates the object

Returns

Return type response

2.20 PhoneNumber

```
class highton.models.phone_number.PhoneNumber (**kwargs)
```

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **location** – fields.StringField(name=HightonConstants.LOCATION)
- **number** – fields.StringField(name=HightonConstants.NUMBER)

ENDPOINT = None

TAG_NAME = 'phone-number'

classmethod decode (*root_element*)

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

static element_from_string (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)

Parses the `xml.etree.ElementTree.Element` to a string

Parameters **element** (`xml.etree.ElementTree.Element`) – a xml element

Returns the parsed string

Return type `str`

encode ()

Encodes the object to a `xml.etree.ElementTree.Element`

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type `dict`

2.21 SubjectData

class `highton.models.subject_data.SubjectData` (***kwargs*)

Variables

- **id** – `fields.IntegerField(name=HightonConstants.ID)`
- **value** – `fields.StringField(name=HightonConstants.VALUE)`
- **type** – `fields.StringField(name=HightonConstants.TYPE)`
- **subject_field_id** – `fields.StringField(name=HightonConstants.SUBJECT_FIELD_ID)`
- **subject_field_label** – `fields.StringField(name=HightonConstants.SUBJECT_FIELD_LABEL)`

ENDPOINT = `None`

TAG_NAME = `'subject_data'`

classmethod **decode** (*root_element*)

Decode the object to the object

Parameters **root_element** (`xml.etree.ElementTree.Element`) – the parsed xml Element

Returns the decoded Element as object

Return type `object`

static **element_from_string** (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type `xml.etree.ElementTree.Element`

static **element_to_string** (*element*)

Parses the `xml.etree.ElementTree.Element` to a string

Parameters `element` (`xml.etree.ElementTree.Element`) – a xml element

Returns the parsed string

Return type `str`

encode()

Encodes the object to a `xml.etree.ElementTree.Element`

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

to_serializable_value()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type `dict`

2.22 Tag

```
class highton.models.tag.Tag(**kwargs)
```

Variables

- **id** – `fields.IntegerField(name=HightonConstants.ID)`
- **name** – `fields.StringField(name=HightonConstants.NAME)`

```
COLLECTION_DATETIME = '%Y%m%d%H%M%S'
```

```
ENDPOINT = 'tags'
```

```
TAG_NAME = 'tag'
```

```
classmethod decode(root_element)
```

Decode the object to the object

Parameters `root_element` (`xml.etree.ElementTree.Element`) – the parsed xml Element

Returns the decoded Element as object

Return type `object`

```
delete()
```

Deletes the object

Returns

Return type `None`

```
static element_from_string(string)
```

Make a Element from a str

Parameters `string` (`str`) – string you want to parse

Returns the parsed xml string

Return type `xml.etree.ElementTree.Element`

```
static element_to_string(element)
```

Parses the `xml.etree.ElementTree.Element` to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode()

Encodes the object to a *xml.etree.ElementTree.Element*

Returns the encoded element

Return type *xml.etree.ElementTree.Element*

classmethod get (*object_id*)

Get all parties (people and companies) associated with a given tag. :param object_id: the primary id of the model :type object_id: integer :return: the parties :rtype: list

classmethod list (*params=None*)

Retrieves a list of the model

Parameters **params** (*dict*) – params as dictionary

Returns the list of the parsed xml objects

Return type list

to_serializable_value()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

2.23 Task

class *highton.models.task.Task* (***kwargs*)

Variables

- **id** – *fields.IntegerField(name=HightonConstants.ID)*
- **recording_id** – *fields.IntegerField(name=HightonConstants.RECORDING_ID)*
- **subject_id** – *fields.IntegerField(name=HightonConstants.SUBJECT_ID)*
- **subject_type** – *fields.StringField(name=HightonConstants.SUBJECT_TYPE)*
- **subject_name** – *fields.StringField(name=HightonConstants.SUBJECT_NAME)*
- **category_id** – *fields.IntegerField(name=HightonConstants.CATEGORY_ID, required=True)*
- **body** – *fields.StringField(name=HightonConstants.BODY, required=True)*
- **frame** – *fields.StringField(name=HightonConstants.FRAME, required=True)*
- **due_at** – *fields.DatetimeField(name=HightonConstants.DUE_AT, required=True)*
- **alert_at** – *fields.DatetimeField(name=HightonConstants.ALERT_AT)*
- **created_at** – *fields.DatetimeField(name=HightonConstants.CREATED_AT)*
- **author_id** – *fields.IntegerField(name=HightonConstants.AUTHOR_ID)*
- **updated_at** – *fields.DatetimeField(name=HightonConstants.UPDATED_AT)*

- **public** – fields.BooleanField(name=HightonConstants.PUBLIC)
- **recurring_period** – fields.StringField(name=HightonConstants.RECURRING_PERIOD)
- **anchor_type** – fields.IntegerField(name=HightonConstants.ANCHOR_TYPE)
- **done_at** – fields.DateTimeField(name=HightonConstants.DONE_AT)
- **owner_id** – fields.IntegerField(name=HightonConstants.OWNER_ID)

COLLECTION_DATETIME = '%Y%m%d%H%M%S'

ENDPOINT = 'tasks'

TAG_NAME = 'task'

complete()
Complete current task :return: :rtype: requests.models.Response

create()
Creates the object

Returns the created object

Return type object

classmethod decode(*root_element*)
Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

delete_tag(*tag_id*)
Deletes a Tag to current object

Parameters **tag_id** (*int*) – the id of the tag which should be deleted

Return type None

static element_from_string(*string*)
Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string(*element*)
Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode()
Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

classmethod `get(object_id)`

Retrieves a single model

Parameters `object_id(integer)` – the primary id of the model

Returns the object of the parsed xml object

Return type object

classmethod `list_all()`

Returns a collection of all tasks visible to the current user.

Returns

Return type list

classmethod `list_assigned()`

Returns a collection of upcoming tasks (tasks that have not yet been completed, regardless of whether they're overdue) that were created by the authenticated user, but assigned to somebody else.

Returns

Return type list

classmethod `list_completed()`

Returns a collection of completed tasks.

Returns

Return type list

classmethod `list_today()`

Returns a collection of uncompleted tasks due for the rest of today for the authenticated user.

Returns

Return type list

classmethod `list_upcoming()`

Returns a collection of upcoming tasks (tasks that have not yet been completed, regardless of whether they're overdue) for the authenticated user

Returns

Return type list

to_serializable_value()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update()

Updates the object

Returns

Return type response

2.24 TaskCategory

class `highton.models.task_category.TaskCategory(**kwargs)`

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **name** – fields.StringField(name=HightonConstants.NAME)
- **color** – fields.StringField(name=HightonConstants.COLOR)
- **account_id** – fields.IntegerField(name=HightonConstants.ACCOUNT_ID)
- **created_at** – fields.DatetimeField(name=HightonConstants.CREATED_AT)
- **updated_at** – fields.DatetimeField(name=HightonConstants.UPDATED_AT)
- **elements_count** – fields.IntegerField(name=HightonConstants.ELEMENTS_COUNT)

COLLECTION_DATETIME = '%Y%m%d%H%M%S'

ENDPOINT = 'task_categories'

TAG_NAME = 'task-category'

create()

Creates the object

Returns the created object

Return type object

classmethod decode (*root_element*)

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

delete()

Deletes the object

Returns

Return type None

static element_from_string (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode()

Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

classmethod `get(object_id)`

Retrieves a single model

Parameters `object_id` (*integer*) – the primary id of the model

Returns the object of the parsed xml object

Return type object

classmethod `list(params=None)`

Retrieves a list of the model

Parameters `params` (*dict*) – params as dictionary

Returns the list of the parsed xml objects

Return type list

to_serializable_value()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

update()

Updates the object

Returns

Return type response

2.25 User

class `highton.models.user.User(**kwargs)`

Variables

- `id` – fields.IntegerField(name=HightonConstants.ID)
- `name` – fields.StringField(name=HightonConstants.NAME)
- `email_address` – fields.StringField(name=HightonConstants.EMAIL_ADDRESS)
- `admin` – fields.BooleanField(name=HightonConstants.ADMIN)
- `token` – fields.StringField(name=HightonConstants.TOKEN)
- `dropbox` – fields.StringField(name=HightonConstants.DROPBOX)
- `created_at` – fields.DatetimeField(name=HightonConstants.CREATED_AT)
- `updated_at` – fields.DatetimeField(name=HightonConstants.UPDATED_AT)

`COLLECTION_DATETIME = '%Y%m%d%H%M%S'`

`ENDPOINT = 'users'`

`TAG_NAME = 'user'`

classmethod `decode(root_element)`

Decode the object to the object

Parameters `root_element` (`xml.etree.ElementTree.Element`) – the parsed xml Element

Returns the decoded Element as object

Return type object

static element_from_string (`string`)
Make a Element from a str

Parameters `string` (`str`) – string you want to parse

Returns the parsed xml string

Return type `xml.etree.ElementTree.Element`

static element_to_string (`element`)
Parses the `xml.etree.ElementTree.Element` to a string

Parameters `element` (`xml.etree.ElementTree.Element`) – a xml element

Returns the parsed string

Return type str

encode ()
Encodes the object to a `xml.etree.ElementTree.Element`

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

classmethod get (`object_id`)
Retrieves a single model

Parameters `object_id` (`integer`) – the primary id of the model

Returns the object of the parsed xml object

Return type object

classmethod list (`params=None`)
Retrieves a list of the model

Parameters `params` (`dict`) – params as dictionary

Returns the list of the parsed xml objects

Return type list

classmethod me ()
Returns information about the currently authenticated user.

Returns

Return type *User*

to_serializable_value ()
Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

2.26 TwitterAccount

```
class highton.models.twitter_account.TwitterAccount (**kwargs)
```

Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **location** – fields.StringField(name=HightonConstants.LOCATION)
- **username** – fields.StringField(name=HightonConstants.USERNAME)
- **url** – fields.StringField(name=HightonConstants.URL)

ENDPOINT = None

TAG_NAME = 'twitter-account'

classmethod decode (*root_element*)

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

static element_from_string (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()

Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

2.27 WebAddress

```
class highton.models.web_address.WebAddress (**kwargs)
```


Variables

- **id** – fields.IntegerField(name=HightonConstants.ID)
- **location** – fields.StringField(name=HightonConstants.LOCATION)
- **url** – fields.StringField(name=HightonConstants.URL)

ENDPOINT = None

TAG_NAME = 'web-address'

classmethod decode (*root_element*)

Decode the object to the object

Parameters **root_element** (*xml.etree.ElementTree.Element*) – the parsed xml Element

Returns the decoded Element as object

Return type object

static element_from_string (*string*)

Make a Element from a str

Parameters **string** (*str*) – string you want to parse

Returns the parsed xml string

Return type xml.etree.ElementTree.Element

static element_to_string (*element*)

Parses the xml.etree.ElementTree.Element to a string

Parameters **element** (*xml.etree.ElementTree.Element*) – a xml element

Returns the parsed string

Return type str

encode ()

Encodes the object to a xml.etree.ElementTree.Element

Returns the encoded element

Return type xml.etree.ElementTree.Element

to_serializable_value ()

Parses the Hightonmodel to a serializable value such dicts, lists, strings This can be used to save the model in a NoSQL database

Returns the serialized HightonModel

Return type dict

3.1 BooleanField

class `highton.fields.boolean_field.BooleanField` (*name*, *required=False*)

The BooleanField represents a field which value will be parsed to Highrise specific boolean format

FALSE = 'false'

MAPPING = {False: 'false', True: 'true'}

TRUE = 'true'

decode (*element*)

Parameters *element* (`xml.etree.ElementTree.Element`) – the element which needs to be parsed

Returns `datetime.date`

Return type the parsed date object

encode ()

Returns

Return type `xml.etree.ElementTree.Element`

to_serializable_value ()

Parse the value to a serializable pythonic value Default: Just return the value

Returns

Return type Any

3.2 DateField

```
class highton.fields.date_field.DateField(name, required=False)
    The DateField represents a field which value will be parsed to Highrise specific date format

    DATE_FORMAT = '%Y-%m-%d'

    decode(element)

        Parameters element (xml.etree.ElementTree.Element) – the element which needs
            to be parsed

        Returns datetime.date

        Return type the parsed date object

    encode()

        Returns

        Return type xml.etree.ElementTree.Element

    to_serializable_value()
        Parse the value to a serializable pythonic value Default: Just return the value

        Returns

        Return type Any
```

3.3 DateTimeField

```
class highton.fields.datetime_field.DatetimeField(name, required=False)

    DATETIME_FORMAT = '%Y-%m-%dT%H:%M:%SZ'

    decode(element)

        Parameters element (xml.etree.ElementTree.Element) – the element which needs
            to be parsed

        Returns the parsed datetime object

        Return type datetime.datetime

    encode()

        Returns the encoded element

        Return type xml.etree.ElementTree.Element

    to_serializable_value()
        Parse the value to a serializable pythonic value Default: Just return the value

        Returns

        Return type Any
```

3.4 Field

class `highton.fields.field.Field` (*name*, *required=False*)

This is an abstract class for a field, which makes the base for an encode and decode method

Variables

- **name** – name of the field which needed to serialize this field to a xml object
- **required** – if the field is required when you create or update an object
- **value** – the value of the field

decode (*element*)

Decodes the value of the element

Parameters *element* –

Returns

Return type

encode ()

Encodes the value of the field and put it in the element also make the check for nil=true if there is one

Returns returns the encoded element

Return type `xml.etree.ElementTree.Element`

to_serializable_value ()

Parse the value to a serializable pythonic value Default: Just return the value

Returns

Return type Any

3.5 IntegerField

class `highton.fields.integer_field.IntegerField` (*name*, *required=False*)

decode (*element*)

Parameters *element* (`xml.etree.ElementTree.Element`) –

Returns the parsed int object

Return type int

encode ()

Returns

Return type `xml.etree.ElementTree.Element`

to_serializable_value ()

Parse the value to a serializable pythonic value Default: Just return the value

Returns

Return type Any

3.6 ListField

class `highton.fields.list_field.ListField`(*name*, *init_class*)

The ListField parses the `init_class` objects in a list

Variables `init_class` – a Highton model

decode (*element*)

Parameters `element` –

Returns the parsed list with `init_class` objects

Return type list

encode ()

Just iterate over the child elements and append them to the current element

Returns the encoded element

Return type `xml.etree.ElementTree.Element`

to_serializable_value ()

Run through the values and parse them to a serializable value

Returns

Return type list

3.7 ObjectField

class `highton.fields.object_field.ObjectField`(*name*, *init_class*)

The ObjectField parses the `init_class` object

Variables `init_class` – a Highton model

decode (*element*)

Decodes the value of the element

Parameters `element` –

Returns

Return type

encode ()

Just encode the object you have as value

Returns the parsed element

Return type `xml.etree.ElementTree.Element`

to_serializable_value ()

Run through all fields of the object and parse the values

Returns

Return type dict

3.8 StringField

class `highton.fields.string_field.StringField`(*name*, *required=False*)

decode (*element*)

Decodes the value of the element

Parameters *element* –

Returns

Return type

encode ()

Encodes the value of the field and put it in the element also make the check for nil=true if there is one

Returns returns the encoded element

Return type `xml.etree.ElementTree.Element`

to_serializable_value ()

Parse the value to a serializable pythonic value Default: Just return the value

Returns

Return type Any

CHAPTER 4

Indices and tables

- `genindex`
- `modindex`
- `search`

h

- `highton.fields.boolean_field`, 47
- `highton.fields.date_field`, 48
- `highton.fields.datetime_field`, 48
- `highton.fields.field`, 49
- `highton.fields.integer_field`, 49
- `highton.fields.list_field`, 50
- `highton.fields.object_field`, 50
- `highton.fields.string_field`, 51
- `highton.models.address`, 3
- `highton.models.associated_party`, 4
- `highton.models.attachment`, 5
- `highton.models.case`, 6
- `highton.models.category`, 9
- `highton.models.comment`, 10
- `highton.models.company`, 11
- `highton.models.contact`, 15
- `highton.models.contact_data`, 16
- `highton.models.deal`, 17
- `highton.models.deal_category`, 21
- `highton.models.email`, 22
- `highton.models.email_address`, 24
- `highton.models.group`, 25
- `highton.models.highton_model`, 27
- `highton.models.instant_messenger`, 27
- `highton.models.note`, 28
- `highton.models.party`, 30
- `highton.models.person`, 32
- `highton.models.phone_number`, 35
- `highton.models.subject_data`, 36
- `highton.models.tag`, 37
- `highton.models.task`, 38
- `highton.models.task_category`, 40
- `highton.models.twitter_account`, 44
- `highton.models.user`, 42
- `highton.models.web_address`, 44

A

add_note() (highton.models.case.Case method), 7
 add_note() (highton.models.company.Company method), 12
 add_note() (highton.models.deal.Deal method), 18
 add_note() (highton.models.person.Person method), 32
 add_tag() (highton.models.case.Case method), 7
 add_tag() (highton.models.company.Company method), 12
 add_tag() (highton.models.deal.Deal method), 18
 add_tag() (highton.models.person.Person method), 33
 Address (class in highton.models.address), 3
 AssociatedParty (class in highton.models.associated_party), 4
 Attachment (class in highton.models.attachment), 5

B

BooleanField (class in highton.fields.boolean_field), 47

C

Case (class in highton.models.case), 6
 Category (class in highton.models.category), 9
 COLLECTION_DATETIME (highton.models.attachment.Attachment attribute), 6
 COLLECTION_DATETIME (highton.models.case.Case attribute), 7
 COLLECTION_DATETIME (highton.models.comment.Comment attribute), 10
 COLLECTION_DATETIME (highton.models.company.Company attribute), 12
 COLLECTION_DATETIME (highton.models.deal.Deal attribute), 18
 COLLECTION_DATETIME (highton.models.deal_category.DealCategory attribute), 21

COLLECTION_DATETIME (highton.models.email.Email attribute), 23
 COLLECTION_DATETIME (highton.models.group.Group attribute), 25
 COLLECTION_DATETIME (highton.models.note.Note attribute), 29
 COLLECTION_DATETIME (highton.models.person.Person attribute), 32
 COLLECTION_DATETIME (highton.models.tag.Tag attribute), 37
 COLLECTION_DATETIME (highton.models.task.Task attribute), 39
 COLLECTION_DATETIME (highton.models.task_category.TaskCategory attribute), 41
 COLLECTION_DATETIME (highton.models.user.User attribute), 42
 Comment (class in highton.models.comment), 10
 COMMENT_OFFSET (highton.models.email.Email attribute), 23
 COMMENT_OFFSET (highton.models.note.Note attribute), 29
 Company (class in highton.models.company), 11
 complete() (highton.models.task.Task method), 39
 Contact (class in highton.models.contact), 15
 ContactData (class in highton.models.contact_data), 16
 create() (highton.models.case.Case method), 7
 create() (highton.models.comment.Comment method), 10
 create() (highton.models.company.Company method), 12
 create() (highton.models.deal.Deal method), 18
 create() (highton.models.deal_category.DealCategory method), 21
 create() (highton.models.email.Email method), 23
 create() (highton.models.group.Group method), 25
 create() (highton.models.note.Note method), 29
 create() (highton.models.person.Person method), 33
 create() (highton.models.task.Task method), 39
 create() (highton.models.task_category.TaskCategory method), 41

D

- DATE_FORMAT (highton.fields.date_field.DateField attribute), 48
- DateField (class in highton.fields.date_field), 48
- DATETIME_FORMAT (highton.fields.datetime_field.DatetimeField attribute), 48
- DatetimeField (class in highton.fields.datetime_field), 48
- Deal (class in highton.models.deal), 17
- DealCategory (class in highton.models.deal_category), 21
- decode() (highton.fields.boolean_field.BooleanField method), 47
- decode() (highton.fields.date_field.DateField method), 48
- decode() (highton.fields.datetime_field.DatetimeField method), 48
- decode() (highton.fields.field.Field method), 49
- decode() (highton.fields.integer_field.IntegerField method), 49
- decode() (highton.fields.list_field.ListField method), 50
- decode() (highton.fields.object_field.ObjectField method), 50
- decode() (highton.fields.string_field.StringField method), 51
- decode() (highton.models.address.Address class method), 3
- decode() (highton.models.associated_party.AssociatedParty class method), 5
- decode() (highton.models.attachment.Attachment class method), 6
- decode() (highton.models.case.Case class method), 7
- decode() (highton.models.category.Category class method), 9
- decode() (highton.models.comment.Comment class method), 10
- decode() (highton.models.company.Company class method), 12
- decode() (highton.models.contact.Contact class method), 15
- decode() (highton.models.contact_data.ContactData class method), 16
- decode() (highton.models.deal.Deal class method), 18
- decode() (highton.models.deal_category.DealCategory class method), 21
- decode() (highton.models.email.Email class method), 23
- decode() (highton.models.email_address.EmailAddress class method), 24
- decode() (highton.models.group.Group class method), 25
- decode() (highton.models.highton_model.HightonModel class method), 27
- decode() (highton.models.instant_messenger.InstantMessenger class method), 28
- decode() (highton.models.note.Note class method), 29
- decode() (highton.models.party.Party class method), 31
- decode() (highton.models.person.Person class method), 33
- decode() (highton.models.phone_number.PhoneNumber class method), 35
- decode() (highton.models.subject_data.SubjectData class method), 36
- decode() (highton.models.tag.Tag class method), 37
- decode() (highton.models.task.Task class method), 39
- decode() (highton.models.task_category.TaskCategory class method), 41
- decode() (highton.models.twitter_account.TwitterAccount class method), 44
- decode() (highton.models.user.User class method), 42
- decode() (highton.models.web_address.WebAddress class method), 45
- delete() (highton.models.case.Case method), 7
- delete() (highton.models.comment.Comment method), 10
- delete() (highton.models.company.Company method), 13
- delete() (highton.models.deal.Deal method), 19
- delete() (highton.models.deal_category.DealCategory method), 21
- delete() (highton.models.email.Email method), 23
- delete() (highton.models.group.Group method), 25
- delete() (highton.models.note.Note method), 29
- delete() (highton.models.person.Person method), 33
- delete() (highton.models.tag.Tag method), 37
- delete() (highton.models.task_category.TaskCategory method), 41
- delete_tag() (highton.models.case.Case method), 7
- delete_tag() (highton.models.company.Company method), 13
- delete_tag() (highton.models.deal.Deal method), 19
- delete_tag() (highton.models.person.Person method), 33
- delete_tag() (highton.models.task.Task method), 39

E

- element_from_string() (highton.models.address.Address static method), 3
- element_from_string() (highton.models.associated_party.AssociatedParty static method), 5
- element_from_string() (highton.models.attachment.Attachment static method), 6
- element_from_string() (highton.models.case.Case static method), 8
- element_from_string() (highton.models.category.Category static method), 9
- element_from_string() (highton.models.comment.Comment static method), 11
- element_from_string() (highton.models.company.Company static method),

method), 41

element_to_string() (highton.models.twitter_account.TwitterAccount static method), 44

element_to_string() (highton.models.user.User static method), 43

element_to_string() (highton.models.web_address.WebAddress static method), 45

Email (class in highton.models.email), 22

EmailAddress (class in highton.models.email_address), 24

EMAILS_OFFSET (highton.models.case.Case attribute), 7

EMAILS_OFFSET (highton.models.company.Company attribute), 12

EMAILS_OFFSET (highton.models.deal.Deal attribute), 18

EMAILS_OFFSET (highton.models.person.Person attribute), 32

encode() (highton.fields.boolean_field.BooleanField method), 47

encode() (highton.fields.date_field.DateField method), 48

encode() (highton.fields.datetime_field.DatetimeField method), 48

encode() (highton.fields.field.Field method), 49

encode() (highton.fields.integer_field.IntegerField method), 49

encode() (highton.fields.list_field.ListField method), 50

encode() (highton.fields.object_field.ObjectField method), 50

encode() (highton.fields.string_field.StringField method), 51

encode() (highton.models.address.Address method), 4

encode() (highton.models.associated_party.AssociatedParty method), 5

encode() (highton.models.attachment.Attachment method), 6

encode() (highton.models.case.Case method), 8

encode() (highton.models.category.Category method), 10

encode() (highton.models.comment.Comment method), 11

encode() (highton.models.company.Company method), 13

encode() (highton.models.contact.Contact method), 16

encode() (highton.models.contact_data.ContactData method), 17

encode() (highton.models.deal.Deal method), 19

encode() (highton.models.deal_category.DealCategory method), 21

encode() (highton.models.email.Email method), 23

encode() (highton.models.email_address.EmailAddress method), 25

encode() (highton.models.group.Group method), 26

encode() (highton.models.highton_model.HightonModel method), 27

encode() (highton.models.instant_messenger.InstantMessenger method), 28

encode() (highton.models.note.Note method), 30

encode() (highton.models.party.Party method), 31

encode() (highton.models.person.Person method), 33

encode() (highton.models.phone_number.PhoneNumber method), 36

encode() (highton.models.subject_data.SubjectData method), 37

encode() (highton.models.tag.Tag method), 38

encode() (highton.models.task.Task method), 39

encode() (highton.models.task_category.TaskCategory method), 41

encode() (highton.models.twitter_account.TwitterAccount method), 44

encode() (highton.models.user.User method), 43

encode() (highton.models.web_address.WebAddress method), 45

ENDPOINT (highton.models.address.Address attribute), 3

ENDPOINT (highton.models.associated_party.AssociatedParty attribute), 5

ENDPOINT (highton.models.attachment.Attachment attribute), 6

ENDPOINT (highton.models.case.Case attribute), 7

ENDPOINT (highton.models.category.Category attribute), 9

ENDPOINT (highton.models.comment.Comment attribute), 10

ENDPOINT (highton.models.company.Company attribute), 12

ENDPOINT (highton.models.contact.Contact attribute), 15

ENDPOINT (highton.models.contact_data.ContactData attribute), 16

ENDPOINT (highton.models.deal.Deal attribute), 18

ENDPOINT (highton.models.deal_category.DealCategory attribute), 21

ENDPOINT (highton.models.email.Email attribute), 23

ENDPOINT (highton.models.email_address.EmailAddress attribute), 24

ENDPOINT (highton.models.group.Group attribute), 25

ENDPOINT (highton.models.highton_model.HightonModel attribute), 27

ENDPOINT (highton.models.instant_messenger.InstantMessenger attribute), 28

ENDPOINT (highton.models.note.Note attribute), 29

ENDPOINT (highton.models.party.Party attribute), 31

ENDPOINT (highton.models.person.Person attribute), 32

ENDPOINT (highton.models.phone_number.PhoneNumber attribute), 35

ENDPOINT (highton.models.subject_data.SubjectData

- attribute), 36
- ENDPOINT (highton.models.tag.Tag attribute), 37
- ENDPOINT (highton.models.task.Task attribute), 39
- ENDPOINT (highton.models.task_category.TaskCategory attribute), 41
- ENDPOINT (highton.models.twitter_account.TwitterAccount attribute), 44
- ENDPOINT (highton.models.user.User attribute), 42
- ENDPOINT (highton.models.web_address.WebAddress attribute), 45
- F**
- FALSE (highton.fields.boolean_field.BooleanField attribute), 47
- Field (class in highton.fields.field), 49
- G**
- get() (highton.models.attachment.Attachment method), 6
- get() (highton.models.case.Case class method), 8
- get() (highton.models.comment.Comment class method), 11
- get() (highton.models.company.Company class method), 13
- get() (highton.models.deal.Deal class method), 19
- get() (highton.models.deal_category.DealCategory class method), 22
- get() (highton.models.email.Email class method), 23
- get() (highton.models.group.Group class method), 26
- get() (highton.models.note.Note class method), 30
- get() (highton.models.person.Person class method), 33
- get() (highton.models.tag.Tag class method), 38
- get() (highton.models.task.Task class method), 39
- get() (highton.models.task_category.TaskCategory class method), 42
- get() (highton.models.user.User class method), 43
- Group (class in highton.models.group), 25
- H**
- highton.fields.boolean_field (module), 47
- highton.fields.date_field (module), 48
- highton.fields.datetime_field (module), 48
- highton.fields.field (module), 49
- highton.fields.integer_field (module), 49
- highton.fields.list_field (module), 50
- highton.fields.object_field (module), 50
- highton.fields.string_field (module), 51
- highton.models.address (module), 3
- highton.models.associated_party (module), 4
- highton.models.attachment (module), 5
- highton.models.case (module), 6
- highton.models.category (module), 9
- highton.models.comment (module), 10
- highton.models.company (module), 11
- highton.models.contact (module), 15
- highton.models.contact_data (module), 16
- highton.models.deal (module), 17
- highton.models.deal_category (module), 21
- highton.models.email (module), 22
- highton.models.email_address (module), 24
- highton.models.group (module), 25
- highton.models.highton_model (module), 27
- highton.models.instant_messenger (module), 27
- highton.models.note (module), 28
- highton.models.party (module), 30
- highton.models.person (module), 32
- highton.models.phone_number (module), 35
- highton.models.subject_data (module), 36
- highton.models.tag (module), 37
- highton.models.task (module), 38
- highton.models.task_category (module), 40
- highton.models.twitter_account (module), 44
- highton.models.user (module), 42
- highton.models.web_address (module), 44
- HightonModel (class in highton.models.highton_model), 27
- I**
- InstantMessenger (class in highton.models.instant_messenger), 27
- IntegerField (class in highton.fields.integer_field), 49
- L**
- list() (highton.models.case.Case class method), 8
- list() (highton.models.company.Company class method), 13
- list() (highton.models.deal.Deal class method), 19
- list() (highton.models.deal_category.DealCategory class method), 22
- list() (highton.models.group.Group class method), 26
- list() (highton.models.person.Person class method), 34
- list() (highton.models.tag.Tag class method), 38
- list() (highton.models.task_category.TaskCategory class method), 42
- list() (highton.models.user.User class method), 43
- list_all() (highton.models.task.Task class method), 40
- list_assigned() (highton.models.task.Task class method), 40
- list_comments() (highton.models.email.Email method), 24
- list_comments() (highton.models.note.Note method), 30
- list_completed() (highton.models.task.Task class method), 40
- list_emails() (highton.models.case.Case method), 8
- list_emails() (highton.models.company.Company method), 14
- list_emails() (highton.models.deal.Deal method), 19
- list_emails() (highton.models.person.Person method), 34
- list_notes() (highton.models.case.Case method), 8

list_notes() (highton.models.company.Company method), 14

list_notes() (highton.models.deal.Deal method), 20

list_notes() (highton.models.person.Person method), 34

list_tags() (highton.models.case.Case method), 9

list_tags() (highton.models.company.Company method), 14

list_tags() (highton.models.deal.Deal method), 20

list_tags() (highton.models.person.Person method), 34

list_tasks() (highton.models.case.Case method), 9

list_tasks() (highton.models.company.Company method), 14

list_tasks() (highton.models.deal.Deal method), 20

list_tasks() (highton.models.person.Person method), 34

list_today() (highton.models.task.Task class method), 40

list_upcoming() (highton.models.task.Task class method), 40

ListField (class in highton.fields.list_field), 50

M

MAPPING (highton.fields.boolean_field.BooleanField attribute), 47

me() (highton.models.user.User class method), 43

N

Note (class in highton.models.note), 28

NOTES_OFFSET (highton.models.case.Case attribute), 7

NOTES_OFFSET (highton.models.company.Company attribute), 12

NOTES_OFFSET (highton.models.deal.Deal attribute), 18

NOTES_OFFSET (highton.models.person.Person attribute), 32

O

ObjectField (class in highton.fields.object_field), 50

OFFSET (highton.models.company.Company attribute), 12

OFFSET (highton.models.deal.Deal attribute), 18

OFFSET (highton.models.person.Person attribute), 32

P

Party (class in highton.models.party), 30

Person (class in highton.models.person), 32

PhoneNumber (class in highton.models.phone_number), 35

S

search() (highton.models.company.Company class method), 14

search() (highton.models.person.Person class method), 34

SEARCH_OFFSET (highton.models.company.Company attribute), 12

SEARCH_OFFSET (highton.models.person.Person attribute), 32

StringField (class in highton.fields.string_field), 51

SubjectData (class in highton.models.subject_data), 36

T

Tag (class in highton.models.tag), 37

TAG_NAME (highton.models.address.Address attribute), 3

TAG_NAME (highton.models.associated_party.AssociatedParty attribute), 5

TAG_NAME (highton.models.attachment.Attachment attribute), 6

TAG_NAME (highton.models.case.Case attribute), 7

TAG_NAME (highton.models.category.Category attribute), 9

TAG_NAME (highton.models.comment.Comment attribute), 10

TAG_NAME (highton.models.company.Company attribute), 12

TAG_NAME (highton.models.contact.Contact attribute), 15

TAG_NAME (highton.models.contact_data.ContactData attribute), 16

TAG_NAME (highton.models.deal.Deal attribute), 18

TAG_NAME (highton.models.deal_category.DealCategory attribute), 21

TAG_NAME (highton.models.email.Email attribute), 23

TAG_NAME (highton.models.email_address.EmailAddress attribute), 24

TAG_NAME (highton.models.group.Group attribute), 25

TAG_NAME (highton.models.highton_model.HightonModel attribute), 27

TAG_NAME (highton.models.instant_messenger.InstantMessenger attribute), 28

TAG_NAME (highton.models.note.Note attribute), 29

TAG_NAME (highton.models.party.Party attribute), 31

TAG_NAME (highton.models.person.Person attribute), 32

TAG_NAME (highton.models.phone_number.PhoneNumber attribute), 35

TAG_NAME (highton.models.subject_data.SubjectData attribute), 36

TAG_NAME (highton.models.tag.Tag attribute), 37

TAG_NAME (highton.models.task.Task attribute), 39

TAG_NAME (highton.models.task_category.TaskCategory attribute), 41

TAG_NAME (highton.models.twitter_account.TwitterAccount attribute), 44

TAG_NAME (highton.models.user.User attribute), 42

TAG_NAME (highton.models.web_address.WebAddress attribute), 45

Task (class in highton.models.task), 38

TaskCategory (class in highton.models.task_category), 40

- to_serializable_value() (high-ton.fields.boolean_field.BooleanField method), 47
- to_serializable_value() (high-ton.fields.date_field.DateField method), 48
- to_serializable_value() (high-ton.fields.datetime_field.DatetimeField method), 48
- to_serializable_value() (high-ton.fields.field.Field method), 49
- to_serializable_value() (high-ton.fields.integer_field.IntegerField method), 49
- to_serializable_value() (high-ton.fields.list_field.ListField method), 50
- to_serializable_value() (high-ton.fields.object_field.ObjectField method), 50
- to_serializable_value() (high-ton.fields.string_field.StringField method), 51
- to_serializable_value() (high-ton.models.address.Address method), 4
- to_serializable_value() (high-ton.models.associated_party.AssociatedParty method), 5
- to_serializable_value() (high-ton.models.attachment.Attachment method), 6
- to_serializable_value() (high-ton.models.case.Case method), 9
- to_serializable_value() (high-ton.models.category.Category method), 10
- to_serializable_value() (high-ton.models.comment.Comment method), 11
- to_serializable_value() (high-ton.models.company.Company method), 14
- to_serializable_value() (high-ton.models.contact.Contact method), 16
- to_serializable_value() (high-ton.models.contact_data.ContactData method), 17
- to_serializable_value() (high-ton.models.deal.Deal method), 20
- to_serializable_value() (high-ton.models.deal_category.DealCategory method), 22
- to_serializable_value() (high-ton.models.email.Email method), 24
- to_serializable_value() (high-ton.models.email_address.EmailAddress method), 25
- to_serializable_value() (high-ton.models.group.Group method), 26
- to_serializable_value() (high-ton.models.highton_model.HightonModel method), 27
- to_serializable_value() (high-ton.models.instant_messenger.InstantMessenger method), 28
- to_serializable_value() (high-ton.models.note.Note method), 30
- to_serializable_value() (high-ton.models.party.Party method), 31
- to_serializable_value() (high-ton.models.person.Person method), 35
- to_serializable_value() (high-ton.models.phone_number.PhoneNumber method), 36
- to_serializable_value() (high-ton.models.subject_data.SubjectData method), 37
- to_serializable_value() (high-ton.models.tag.Tag method), 38
- to_serializable_value() (high-ton.models.task.Task method), 40
- to_serializable_value() (high-ton.models.task_category.TaskCategory method), 42
- to_serializable_value() (high-ton.models.twitter_account.TwitterAccount method), 44
- to_serializable_value() (high-ton.models.user.User method), 43
- to_serializable_value() (high-ton.models.web_address.WebAddress method), 45
- TRUE (high-ton.fields.boolean_field.BooleanField attribute), 47
- TwitterAccount (class in high-ton.models.twitter_account), 44
- ## U
- update() (high-ton.models.case.Case method), 9
- update() (high-ton.models.comment.Comment method), 11
- update() (high-ton.models.company.Company method), 14
- update() (high-ton.models.deal.Deal method), 20
- update() (high-ton.models.deal_category.DealCategory method), 22
- update() (high-ton.models.email.Email method), 24
- update() (high-ton.models.group.Group method), 26
- update() (high-ton.models.note.Note method), 30
- update() (high-ton.models.person.Person method), 35
- update() (high-ton.models.task.Task method), 40

`update()` (highton.models.task_category.TaskCategory method), [42](#)
`update_status()` (highton.models.deal.Deal method), [20](#)
`User` (class in highton.models.user), [42](#)

W

`WebAddress` (class in highton.models.web_address), [44](#)