simula$_{l}ibrestclientDocumentation$

Release 1.0.5

Espen Angell Kristiansen

Sep 27, 2017

Contents

CHAPTER 1

Documentation

simula_bibrestclient.client — REST client

class simula_bibrestclient.client.**BibFolder**(*username*, *password*, *folderurl='http://plonerest.simula.no/publications/'*, ***kwargs*)

Bases: *simula_bibrestclient.client.BibResource*

Bibliography folder REST API.

bulk_update(*items*, *pretend=False*)

Perform HTTP PUT to update many items.

Note: This is a very thin wrapper around *put()*. See the source for this method (link on the right hand side) to see what it does.

Parameters **items** – List of items to update.

Pretend Just pretend to make the changes? If `True`, no changes are made to the database, however the response will still be just as if we used `pretend=False`.

create_item(*id*, *portal_type*, *attributes*, *pretend=False*)

Perform HTTP POST to create a bibliography item within this folder.

Note: This is a very thin wrapper around *post()*. See the source for this method (link on the right hand side) to see what it does.

Parameters

- **id** – ID (short name) of the new item.
- **portal_type** – The portal_type of the bibliography item.

- **attributes** – Attributes for the new item. These are the same as the `attributes` documented in the REST API docs. Note that `attributes['id']` is ignored if included (The `id` parameter is used instead).

Pretend Just pretend to make the changes? If `True`, no changes are made to the database, however the response will still be just as if we used `pretend=False`.

get_restapi_url()
Returns the REST API URL of this bibfolder. When this URL is opened in a browser, the user sees the docs for the REST API.

get_website_url()
Returns the URL for this bibfolder on the website (not the REST API url).

search(*simula_author_usernames=[]*, *author_names=[]*, *search=''*, *itemids=[]*)
Perform HTTP GET to search the publication catalog. The default is to return all publications, however the results can be limited using the parameters below. Only the first search limiting parameter that is not `bool(False)` is used, and they are checked in the order they are listed.

Parameters

- **simula_author_usernames** – List of author Plone usernames. Matches exact usernames (no fuzzy searching). E.g.: `['hpl', 'griff']`
- **author_names** – List of author names. E.g.: `['Griwodz C', 'Langtangen H']`
- **search** – Free text search. Searches the SearchableText index. E.g: `"addresses the problem of managing an evolving"`
- **itemids** – Search for one or more item IDs. Matches the given itemids exactly (no fuzzy searching). E.g.: `['Simula.010', 'Simula.002']`

class `simula_bibrestclient.client.`**BibItem**(*itemid*, *username*, *password*, *folderurl='http://plonerest.simula.no/publications/'*, ***kwargs*)
Bases: *simula_bibrestclient.client.BibResource*

Bibliography item (e.g.: ArticleReferece, PhdThesis, Book, ...) REST API.

Note: Why is portal_type a method argument instead of a constructor argument?

Because `get` does not require `portal_type`, and we need to be able to perform a `get`-request to find the `portal_type`. We do not store any state in the class, so `portal_state` is not automatically maintained internally (since this would require an extra get-request for each update).

Parameters **itemid** – The id of the bibliography item.

browser_confirm_delete()
Open a webbrowser tab on the *get_browser_deleteurl()*. This view asks the user to confirm if they want to delete the item or not.

classmethod decode_pdf(*simula_pdf_file_dict*)
The reverse of *encode_pdf()*.

classmethod decode_pdfdata(*base64data*)
Decode `base64data`.

classmethod encode_pdf(*filename*, *data*, *content_type='application/pdf'*)
Create a `simula_pdf_file`-compatible dict.

Parameters

- **filename** – The name of the file (E.g.: `"myfile.pdf"`). This file is not read from disk, the string is the filename that users get when they download the PDF from the website.
- **data** – A string containing the data of the PDF.
- **content_type** – The content-type of the file. Defaults to "application/pdf".

Returns A `simula_pdf_file`-compatible dict where `data` is base64 encoded.

classmethod encode_pdfdata(*data*)
Returns the `data`-string base64-encoded.

classmethod encode_pdffile(*filepath*, *content_type='application/pdf'*)
Create a `simula_pdf_file`-compatible dict from the file at the given `filepath`. Reads the file from disk and uses *encode_pdf()*.

Parameters

- **filepath** – Path to a file on the local filesystem.
- **content_type** – The content-type of the file. Defaults to "application/pdf".

get(*include_filefields=False*)
Perform HTTP GET to get the bibliography item.

get_browser_deleteurl()
Returns the URL for the confirm-delete view for this bibliography item on the website.

get_restapi_url()
Returns the REST API URL of this bibliography item. When this URL is opened in a browser, the user sees the docs for the REST API.

get_website_url()
Returns the URL for this bibliography item on the website (not the REST API url).

publish_internally(*portal_type*)
Shortcut for `update(portal_state='publish_internally')`.

request_approval_from_coauthors(*portal_type*)
Shortcut for `update(portal_state='submit')`.

update(*portal_type*, *portal_state=None*, *attributes={}*, *pretend=False*)
Perform HTTP PUT to update the item bibliography item.

Note: This is a very thin wrapper around *put()*. See the source for this method (link on the right hand side) to see what it does.

Parameters

- **portal_state** – Change the state of the bibitem. When an item is created, its state is `private`. Any request changing or getting the item includes `portal_state_transitions`, which describes the next possible states.
- **portal_type** – The portal_type for the bibliography item.
- **attributes** – Attributes to update. These are the same as the `attributes` documented in the REST API docs.

Pretend Just pretend to make the changes? If `True`, no changes are made to the database, however the response will still be just as if we used `pretend=False`.

class `simula_bibrestclient.client.`**BibResource**(*url*, *username*, *password*, *mimetype='application/json'*, *decode_output=False*)

Bases: `restkit.resource.Resource`

Base class for the REST APIs.

Parameters

- **url** – URL to the REST API (including @@rest).
- **username** – Simula website username.
- **password** – Simula website password.
- **mimetype** – See *mimetype*.
- **decode_output** – See *decode_output*.

decode(*data*)
If *decode_output* is True, decode decode the response and return a Python datastructure. If *decode_output* is False, return `data`.

encode(*data*)
JSON-encode `data`.

post(*requestdata*)
Perform HTTP POST request with `requestdata` in the request body and *mimetype* in the `Accept` header.

Parameters **requestdata** – Bytestring encoded with as *mimetype*.

put(*requestdata*)
Perform HTTP PUT request with `requestdata` in the request body and *mimetype* in the `Accept` header.

Parameters **requestdata** – Bytestring encoded with as *mimetype*.

decode_output = None
Decode the response from the server? If not `mimetype=='application/json'`, this will raise an exception if `True`.

mimetype = None
The mimetype to use for input/output

simula_bibrestclient.diff — Diff utils

Helpers for generating a diff of two python objects.

`simula_bibrestclient.diff.`**create_diff**(*pydataA*, *pydataB*)
Return a string containing the diff of `pydataA` and `pydataB`.

`simula_bibrestclient.diff.`**create_stringdiff**(*stringA*, *stringB*)
Return a string containing the diff of `stringA` and `stringB`.

simula_bibrestclient.cli — Command line interface

simula_bibrestclient.cli.main.**main**(*arguments=None*, *subcommands=[]*)

The simula_bibrestclient command.

Parameters

- **arguments** – Command-line arguments as a list, excluding the program name. Defaults to sys.args[1:].
- **subcommands** – List of Command-subclasses.

class simula_bibrestclient.cli.search.**Search**(*args*, *auth*)

Bases: simula_bibrestclient.cli.command.Command

help = 'Search for bibliographies.'

Help for the command.

name = 'search'

Name of the command.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

Index

U