sdklib Documentation

Release 1.12.0

Ivan Martin Vedriel

Jan 14, 2022

Contents

Sdklib helps you to write you own client library which will consume a specific service.

CHAPTER 1

Highlights

- Python 2.7+ or 3.3+.
- Only http/https protocol is currently supported.
- BDD integration.

CHAPTER 2

Install

Install the `sdklib` package using pip:

```
pip install sdklib
```

CHAPTER 3

Sample

Find my first SDK on github: https://github.com/ivanprjcts/my-first-sdk

```
from sdklib.http import HttpSdk

class FirstSdk(HttpSdk):
    """
    My First Sdk.
    """
    DEFAULT_HOST = "http://mockapi.sdklib.org"

    API_ITEMS_URL_PATH = "/items/"

    def create_item(self, name, description=None):
        """
        Create an item.

        :type name: str
        :type description: str
        :return: SdkResponse
        """
        params = parse_args(name=name, description=description)
        return self.post(self.API_ITEMS_URL_PATH, body_params=params)
```

CHAPTER 4

Run tests

Running testing with coverage:

```
py.test --cov=sdklib tests/
```

CHAPTER 5

Contributing

1. Check for open issues or open a fresh issue to start a discussion around a feature idea or a bug.
2. Fork the repository on GitHub to start making your changes to the master branch (or branch off of it).
3. Write a test which shows that the bug was fixed or that the feature works as expected.
4. Send a pull request and bug the maintainer until it gets merged and published. Make sure to add yourself to Authors.

CHAPTER 6

Authors

- Ivan Martin Vedriel - @ivanprjcts
- Rubén González Alonso - @rgonalo

6.1 Writing your First SDK

6.1.1 Introduction

This tutorial will cover creating a simple SDK for managing a REST-ful API.

Note: The code for this tutorial is available in the ivanprjcts/my-first-sdk repository on GitHub.

6.1.2 Setting up a new environment

Before we do anything else we'll create a new virtual environment, using virtualenv. This will make sure our package configuration is kept nicely isolated from any other projects we're working on.

```
virtualenv env
source env/bin/activate
```

Now that we're inside a virtualenv environment, we can install our package requirements.

```
pip install sdklib
```

Note: To exit the virtualenv environment at any time, just type *deactivate*.

6.1.3 Getting started

Okay, we're ready to get coding. To get started, let's create a new project to work with.

```
cd ~
mkdir my-first-sdk
cd my-first-sdk
```

Once that's done we can create project structure that we'll use to create a simple Web API Client SDK.

```
mkdir first_sdk tests
touch README.md first_sdk/__init__.py first_sdk/first_sdk.py tests/__init__.py tests/
↪test_first_sdk.py
tree
.
├── README.md
├── first_sdk
│   ├── __init__.py
│   └── first_sdk.py
└── tests
    ├── __init__.py
    └── test_first_sdk.py
```

6.1.4 My First SDK

We'll need to edit our *first_sdk/first_sdk.py* file:

```
from sdklib.http import HttpSdk
from sdklib.util.parser import safe_add_end_slash, parse_args


class FirstSdk(HttpSdk):
    """
    My First Sdk.
    """
    DEFAULT_HOST = "http://mockapi.sdklib.org"

    API_ITEMS_URL_PATH = "/items/"

    def create_item(self, name, description=None):
        """
        Create an item.
        :param name: str
        :param description: str (optional)
        :return: SdkResponse
        """
        params = parse_args(name=name, description=description)
        return self.post(self.API_ITEMS_URL_PATH, body_params=params)

    def get_items(self, item_id=None):
        """
        Retrieve all items if 'item_id' is None. Otherwise, get specified item by
↪'item_id'.
        :param item_id: str (optional)
        :return: SdkResponse
        """
        return self.get(self.API_ITEMS_URL_PATH + safe_add_end_slash(item_id))

    def update_item(self, item_id, name, description=None):
        """
```

(continues on next page)

(continued from previous page)

```
        Update an item.
        :param item_id: str
        :param name: str
        :param description: str (optional)
        :return: SdkResponse
        """
        params = parse_args(name=name, description=description)
        return self.put(self.API_ITEMS_URL_PATH + item_id + '/', body_params=params)

    def delete_item(self, item_id):
        """
        Remove an item.
        :param item_id: str
        :return: SdkResponse
        """
        return self.delete(self.API_ITEMS_URL_PATH + item_id + '/')
```

Okay, we're ready to test.

6.1.5 Testing my First SDK

Let's edit our *tests/test_first_sdk.py* file:

```
import unittest

from first_sdk.first_sdk import FirstSdk


class TestFirstSdk(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        cls.api = FirstSdk()

    @classmethod
    def tearDownClass(cls):
        pass

    def test_crud_items(self):
        """
        Test the creation, reading, update and deletion of an item.
        """
        response = self.api.create_item("ItemName", "Some description")
        self.assertEqual(response.status, 201)

        item_id = response.data["pk"]
        self.assertEqual("ItemName", response.data["name"])
        self.assertEqual("Some description", response.data["description"])

        response = self.api.get_items()
        self.assertEqual(response.status, 200)
        self.assertIn("results", response.data)
        self.assertTrue(isinstance(response.data["results"], list))

        response = self.api.get_items(item_id)
```

(continues on next page)

(continued from previous page)

```
        self.assertEqual(response.status, 200)
        self.assertEqual("ItemName", response.data["name"])
        self.assertEqual("Some description", response.data["description"])

        response = self.api.update_item(item_id, "New name")
        self.assertEqual(response.status, 200)
        self.assertEqual("New name", response.data["name"])
        self.assertEqual("Some description", response.data["description"])

        response = self.api.delete_item(item_id)
        self.assertEqual(response.status, 204)
```

6.2 Configuration

6.2.1 Secure HTTP

By default, ssl certificates are not validated.

6.2.2 Default Configuration

Parameter	Default value
DEFAULT_HOST	http://127.0.0.1:80
DEFAULT_PROXY	None
DEFAULT_RENDERER	JSONRenderer()

DEFAULT_HOST

Value: "http://127.0.0.1:80"

A string that will be used as host default value.

- It can be also modified by using set_default_host() class method.

DEFAULT_PROXY

Value: None (no proxy)

A string that will be used as proxy default value.

- It can be also modified by using set_default_proxy() class method.

DEFAULT_RENDERER

Value: JSONRenderer() object

A renderer object that will be used to build the body for any request.

For more in depth information, see *Renderers*.

6.2.3 URLS

host

Default: *DEFAULT_HOST*

A string that will be automatically included at the beginning of the url generated for doing each http request.

prefix_url_path

Default: "" (Empty string)

A string that will be automatically prepended to all urls.

url_path_params

Default: {} (Empty dictionary)

A dictionary mapping strings to string format that take a model object and return the generated URL. It works like string format with dictionary:

```
"/path/to/{project_id}/{lang}".format(**{"project_id": 1, "lang": "es"})
```

url_path_format

Default: None

A string that will be automatically included (suffixed) to all urls. For example:

```
.json or .xml
```

6.2.4 authentication_instances

Default: () (Empty tuple)

List (or tuple) of authentication objects that will be used for building the request.

For more in depth information, see *Authentication*.

6.2.5 response_class

Default: HttpResponse

For more in depth information, see *Response*.

6.2.6 ssl_verify

Default: None (undefined), validate certs (True)

Define if certificates are required for the SSL connection. They will be validated, and if validation fails, the connection will also fail.

Values: (bool) *True* or ‘False

This value could also be defined using environment environment variable *SDKLIB_SSL_VERIFY*

```
export SDKLIB_SSL_VERIFY=False
```

6.3 Renderers

Renderers are the managers of request body enconding and the content-type header.

Renderer name	Content-type	Encoding
form	application/x-www-form-urlencoded	param1=value1&param2=value2
multipart	multipart/form-data	Content-Disposition: form-data; name=”param1”nnvalue1
plain	text/plain; charset=utf-8	param1=value1\nparam2=value2
json	application/json	{“param1”: “value1”, “param2”: “value2”}

6.3.1 JSONRenderer

Build the body for a *application/json* request.

6.3.2 FormRenderer

Build the body for a *application/x-www-form-urlencoded* request.

6.3.3 MultiPartRenderer

Build the body for a *multipart/form-data* request.

6.3.4 PlainTextRenderer

Build the body for a *text/plain* request.

6.3.5 Renderers module

class sdklib.http.renderers.**BaseRenderer**
Bases: object

DEFAULT_CONTENT_TYPE = ''

encode_params(*data=None*, ***kwargs*)
Build the body for a request.

class sdklib.http.renderers.**CustomRenderer**(*content_type*)
Bases: *sdklib.http.renderers.BaseRenderer*

encode_params(*data=None*, ***kwargs*)
Build the body for a custom request.

class sdklib.http.renderers.**FormRenderer**(*collection_format='multi'*, *output_str='javascript'*, *sort=False*)

Bases: *sdklib.http.renderers.BaseRenderer*

COLLECTION_SEPARATORS = {'csv': ',', 'pipes': '|', 'ssv': ' ', 'tsv': '\t'}

DEFAULT_CONTENT_TYPE = 'application/x-www-form-urlencoded'

VALID_COLLECTION_FORMATS = ['multi', 'csv', 'ssv', 'tsv', 'pipes', 'encoded']

collection_format

encode_params(*data=None*, ***kwargs*)

Encode parameters in a piece of data. Will successfully encode parameters when passed as a dict or a list of 2-tuples. Order is retained if data is a list of 2-tuples but arbitrary if parameters are supplied as a dict.

class sdklib.http.renderers.**JSONRenderer**

Bases: *sdklib.http.renderers.BaseRenderer*

DEFAULT_CONTENT_TYPE = 'application/json'

encode_params(*data=None*, ***kwargs*)

Build the body for a application/json request.

class sdklib.http.renderers.**MultiPartRenderer**(*boundary='———-ThIs_Is_tHe_bouNdaRY'*, *output_str='javascript'*)

Bases: *sdklib.http.renderers.BaseRenderer*

encode_params(*data=None*, *files=None*, ***kwargs*)

Build the body for a multipart/form-data request. Will successfully encode files when passed as a dict or a list of tuples. Order is retained if data is a list of tuples but arbitrary if parameters are supplied as a dict. The tuples may be string (filepath), 2-tuples (filename, fileobj), 3-tuples (filename, fileobj, contentype) or 4-tuples (filename, fileobj, contentype, custom_headers).

class sdklib.http.renderers.**PlainTextRenderer**(*charset=None*, *collection_format='multi'*, *output_str='javascript'*)

Bases: *sdklib.http.renderers.BaseRenderer*

COLLECTION_SEPARATORS = {'csv': ',', 'pipes': '|', 'ssv': ' ', 'tsv': '\t'}

VALID_COLLECTION_FORMATS = ['multi', 'csv', 'ssv', 'tsv', 'pipes', 'plain']

collection_format

encode_params(*data=None*, ***kwargs*)

Build the body for a text/plain request. Will successfully encode parameters when passed as a dict or a list of 2-tuples. Order is retained if data is a list of 2-tuples but arbitrary if parameters are supplied as a dict.

get_content_type(*charset=None*)

class sdklib.http.renderers.**XMLRenderer**

Bases: *sdklib.http.renderers.BaseRenderer*

DEFAULT_CONTENT_TYPE = 'application/xml'

encode_params(*data=None*, ***kwargs*)

Build the body for a application/xml request.

sdklib.http.renderers.**get_primitive_as_string**(*strings_dict*, *value*)

sdklib.http.renderers.**get_renderer**(*name=None*, *mime_type=None*)

sdklib.http.renderers.**guess_file_name_stream_type_header**(*args*)
Guess filename, file stream, file type, file header from args.

Parameters **args** – may be string (filepath), 2-tuples (filename, fileobj), 3-tuples (filename, fileobj, contentype) or 4-tuples (filename, fileobj, contentype, custom_headers). :return: filename, file stream, file type, file header

sdklib.http.renderers.**to_string**(*value*, *lang='javascript'*)

sdklib.http.renderers.**url_encode**(*params*, *sort=False*)

6.4 Authentication

6.4.1 Session authentication

Session authentication is mostly used for AJAX clients that are running in the same session context as a website.

A client usually authenticates with its credentials and receives a session_id (which can be stored in a cookie) and attaches this to every subsequent outgoing request. So this could be considered a "token" as it is the equivalent of a set of credentials. It is just an identifier and the server does everything else.

HttpSdk class implements a mechanism to save automatically cookies embedded into *SET-COOKIE* header in each response.

If a cookie object is stored locally, then every request will contain a *COOKIE* header with this value.

Session module

class sdklib.http.session.**Cookie**(*headers=None*)
Bases: object

Wrapper of python Cookie class.

See https://docs.python.org/2/library/cookie.html

as_cookie_header_value()

get(*key*, *default=None*)

getcookie()

is_empty()

items()

load_from_headers(*headers*)

update(*cookie*)

Login method

Sdklib provides an abstract method for login purposes.

- *.login(self, **kargs)*

Do a *POST* request to *LOGIN_URL_PATH* using parameters passed to method.

CRSR Token

Sometimes, you'll need to make sure you include a valid CSRF token for any "unsafe" HTTP method calls, such as PUT, PATCH, POST or DELETE requests.

6.4.2 Basic authentication

HTTP Basic authentication (BA) implementation is the simplest technique for enforcing access controls to web resources because it doesn't require cookies, session identifiers, or login pages; rather, HTTP Basic authentication uses standard fields in the HTTP header, obviating the need for handshakes.

The Authorization header

The Authorization field is constructed as follows:

1. The username and password are combined with a single colon.
2. The resulting string is encoded using the RFC2045-MIME variant of Base64, except not limited to 76 char/line.
3. The authorization method and a space i.e. "Basic " is then put before the encoded string.

For example,

```
Authorization: Basic QWxhZGRpbjpPcGVuU2VzYW1l
```

6.4.3 11Paths authentication

All requests must be signed. The signing process is a simplified version of the 2-legged Oauth protocol.

Every HTTP request to the API must be accompanied by two authentication headers: Authorization and Date.

The Authorization header

The Authorization header must have the following format:

```
11PATHS requestId requestSignature
```

- **requestId** is an alphanumeric identifier obtained when registering a new API.
- **11PATHS** is a constant that determines the Authentication method.
- **applicationId** is an alphanumeric identifier obtained when registering a new application.
- **requestSignature** is a signature derived from the url, parameters, custom headers and date of the current request, all hashed using a secret that is also obtained with the applicationId when registering the application.

The request signature is a base64 encoded and HMAC-SHA1 signed string. The string must be created following this process.

1. Start with an empty string.
2. Append the uppercase method name. Currently, only the values GET, POST, PUT, DELETE are supported.
3. Append a line separator, " " (unicode point U+000A).

4. Append a string with the current date in the exact format yyyy-MM-dd HH:mm:ss. Everything in the format should be self explanatory, noting that everything is numeric and should be zero-padded if needed so that all numbers have two digits except the year, having four. This value must be kept to be used in the Date header. The signature checking process will fail if both don't match.
5. Append a line separator, " " (unicode point U+000A).
6. Serialize all headers specific to this application (not every HTTP header in the request). These headers all have their names starting with X-11paths-.
    a. Convert all header names to lower case.
    b. Order the headers by header name in alphabetical ascending order.
    c. For every header value, convert multiline headers into single line by replacing newline characters " " by single spaces " ".
    d. Create an empty string. Then, for every header after the order and transformations above, add to the newly created string the header name followed by a colon ":" and followed by the header value. Each name:value should be separated from the next by a single space " ".
    e. Trim the string to make sure it doesn't contain any spacing characters at the beginning or end.
7. Append a line separator, " " (unicode point U+000A).
8. Append the url encoded query string consisting on the path (starting with the first forward slash) and the query parameters. The query string must not contain the host name or port, and must not contain any spacing characters prefixing or suffixing it.
9. Only for POST or PUT requests, attach a line separator, " " (Unicode Point U+000A).
10. Only for POST or PUT requests, serialize the request parameters as follows, the name of the parameter and its value, the UTF-8 representation of its URL coding.
    a. Order the parameters by parameter name in ascending alphabetical order and then by parameter value.
    b. Create an empty chain. Then, for each parameter after ordering, add to the newly created chain the parameter name followed by an equal sign "=" and the value of the parameter. Each name=value should be separated from the next by an ampersand "&".
    c. Trim the string to make sure it doesn't contain any spacing characters at the beginning or end.

Once the string has been created following the process described above, it must be signed using the HMAC-SHA1 algorithm and the secret that was obtained when registering the application. After signing, its raw binary data must be encoded in base64. The resulting string is the requestSignature to be added to Authorization header.

The X-11Paths-Date header

The **X-11Paths-Date** header contains the value of the current UTC date and must have the following format:

```
yyyy-MM-dd HH:mm:ss
```

- **yyyy** is the year.
- **MM** is the number of month.
- **dd** is the number of day.
- **HH** is the hour in 24h format.
- **mm** is the minute within the hour and ss is the second within the minute.

All values must be zero-padded so that they are all 2 digit values except for the year which is 4.

It is very important that the value and format of this header is the exact same used in the process of creating the **requestSignature** for the authorization header as explained above.

Note you can still use the standard HTTP Header **Date** in whichever format you want, such as RFC 1123. Just make sure to not confuse both and always use the value you use in **X-11Paths-Date** in the signature process. The API will ignore the standard **Date** header.

Authorization module

class sdklib.http.authorization.**AbstractAuthentication**
Bases: object

apply_authentication(*context*)

class sdklib.http.authorization.**BasicAuthentication**(*username*, *password*)
Bases: *sdklib.http.authorization.AbstractAuthentication*

apply_authentication(*context*)

class sdklib.http.authorization.**X11PathsAuthentication**(*app_id*, *secret*, *utc=None*)
Bases: *sdklib.http.authorization.AbstractAuthentication*

apply_authentication(*context*)

sdklib.http.authorization.**basic_authorization**(*username*, *password*)

sdklib.http.authorization.**x_11paths_authorization**(*app_id*, *secret*, *context*, *utc=None*)
Calculate the authentication headers to be sent with a request to the API.

Parameters

- **app_id** –
- **secret** –

:param context :param utc: :return: array a map with the Authorization and Date headers needed to sign a Latch API request

6.5 Response

class sdklib.http.response.**AbstractBaseHttpResponse**(*resp*)
Bases: object

Wrapper of Urllib3 HTTPResponse class needed to implement any HttpSdk response class.

See Urllib3.

cookie

headers
Returns a dictionary of the response headers.

urllib3_response = None

class sdklib.http.response.**Api11PathsResponse**(*resp*)
Bases: *sdklib.http.response.AbstractBaseHttpResponse*, *sdklib.http.response.JsonResponseMixin*

This class models a response from any of the endpoints in most of 11Paths APIs.

It consists of a "data" and an "error" elements. Although normally only one of them will be present, they are not mutually exclusive, since errors can be non fatal, and therefore a response could have valid information in the data field and at the same time inform of an error.

data

Returns data part of the API response into a dictionary

error

@return Error the error part of the API response, consisting of an error code and an error message

class sdklib.http.response.**Error**(*json_data*)

Bases: object

code

json

message

class sdklib.http.response.**HttpResponse**(*resp*)

Bases: *sdklib.http.response.Response*, *sdklib.http.response.AbstractBaseHttpResponse*

Wrapper of Urllib3 HTTPResponse class compatible with AbstractBaseHttpResponse.

See Urllib3.

reason

class sdklib.http.response.**JsonResponseMixin**

Bases: object

case_insensitive_dict

json

class sdklib.http.response.**Response**(*headers=None*, *status=None*, *status_text=None*, *http_version=None*, *body=None*)

Bases: *sdklib.http.response.JsonResponseMixin*

body

data

headers

Returns a dictionary of the response headers.

html

Returns HTML response data.

http_version

raw

Returns urllib3 response data.

status

status_text

xml

6.6 License

—

6.6.1 Included projects

- Urllib3 - View license.
- Xmltodict - View license.

Many thanks to the authors and contributors of those projects.

6.6.2 Sdklib License (BSD)

6.7 Release Notes

6.7.1 Upgrading

To upgrade Sdklib to the latest version, use pip:

```
pip install -U sdklib
```

6.7.2 Sdklib 1.12.x series

Sdklib 1.12.0

- Support Python 3.10.

6.7.3 Sdklib 1.11.x series

Sdklib 1.11.2

- Fix ssl_verify behaviour for old versions of urllib3.

Sdklib 1.11.1

- Fix 11paths authorization using body in python3.

Sdklib 1.11.0

- Allow ssl_verify config param.

6.7.4 Sdklib 1.10.x series

Sdklib 1.10.5

- Fix multipart boundary characters according to RFC (https://www.w3.org/Protocols/rfc1341/7_2_Multipart.html).

Sdklib 1.10.3

- New experimental feature: support socks proxy protocol.

Sdklib 1.10.2

- Fix bug.

Sdklib 1.10.1

- Improve getting text from lxml elements.

Sdklib 1.10.0

- Fix bug setting cookies after receiving several "Set-Cookie" headers.
- Update urllib3 version.
- Remove experimental support to proxy socks protocol.

6.7.5 Sdklib 1.9.x series

Sdklib 1.9.6

- Support proxy socks protocol.

Sdklib 1.9.5

- Fix installation.

Sdklib 1.9.4

- Move ignore warnings function to shortcuts.

Sdklib 1.9.3

- Allow to ignore warnings.
- Fix bug in getparent htmlElem method.

Sdklib 1.9.2

- Fix bug in HAR requests (urlencoded params were encoded twice).
- Add 'timeout' attribute to http_request_context.

Sdklib 1.9.1

- Add getparent method to LxmlElem class.

Sdklib 1.9.0

- Put 'incognito_mode' as class attribute.
- Add cache decorator for python 3+.

6.7.6 Sdklib 1.8.x series

Sdklib 1.8.11

- Fix 11paths authorization bug in 'POST'/'PUT' json requests without body.
- Headers as case insensitive dictionary.

Sdklib 1.8.10

- Fix 11paths authorization bug in requests with query params.

Sdklib 1.8.9

- Fix 11paths authorization bug in some 'POST', 'PUT' requests.

Sdklib 1.8.8

- Create HTMLElem including AbstractBaseHTML functionality.
- Add HAR objects.
- Add cookie property to HttpRequestContext.
- Do not add *X_11PATHS_BODY_HASH* header when body is empty.
- Refactor HttpResponse.
- Add find_by_name method to html objects.

Sdklib 1.8.7

- Update cookies rather than replace them.
- Fix "The HTTP reason phrase should not be" behave step.
- Add "The HTTP reason phrase should contain" behave step.
- Add BaseHttpResponse class.
- Fix some bugs (#50).

Sdklib 1.8.6

- Fix 11paths authorization bug.
- Add insensitive (sort) parameter to to_key_val_list function.

Sdklib 1.8.5.3

- Fix behave steps bug.

Sdklib 1.8.5.2

- Fix Api11PathsResponse bug.

Sdklib 1.8.5.1

- Fix Api11PathsResponse bug.

Sdklib 1.8.5

- Create AbstractHttpResponse class.
- Remove some properties from Api11PathsResponse.

Sdklib 1.8.4

- Make get Api11PathsResponse data, error, code and message case insensitive.
- Add CaseInsensitiveDict class.
- Fix some bugs.

Sdklib 1.8.3

- Add behave steps.
- Fix some bugs.
- Separate requirements_dev.txt into different files.

Sdklib 1.8.2

- Add Api11PathsResponse.

Sdklib 1.8.1

- Add guess_file_name_stream_type_header() method.
- Fix 11paths auth bug.

Sdklib 1.8

- Add test coverage reports.
- Add some tests.
- Remove rrserver.
- Remove behave api steps.
- Remove unused modules.
- Fix some bugs.

6.7.7 Sdklib 1.7.x series

Sdklib 1.7.2

- Fix some bugs.

Sdklib 1.7.1

- Fix some bugs.

Sdklib 1.7

- Return more parameters into urlsplit function.
- Add generate_url() function.
- Add lxml as optional requirement.
- Support xpath functions such as contains() using lxml.

6.7.8 Sdklib 1.6.x series

Sdklib 1.6.6

- Allow to redirect http requests.

Sdklib 1.6.5

- Use an internal logger instance to print request and response logs.
- Add clear method to http request context.
- Add fields_to_clear attribute to http request context.

Sdklib 1.6

- Custom content-type header has priority over renderer content-type.
- Get update_content_type parameter from context.
- Add BaseRenderer.
- Add CustomRenderer.

6.7.9 Sdklib 1.5.x series

Sdklib 1.5.2

- Add manifest.

Sdklib 1.5.1

- Fix requirements.

Sdklib 1.5

- Add HTML parsed response.

6.7.10 Sdklib 1.4.x series

Sdklib 1.4.2

- Fix bug: 11paths authorization header is not correct using multiples form params.

Sdklib 1.4.1

- Fix bug: ensure url path params is never None.

Sdklib 1.4

- Add XMLRenderer interface.
- Add json property to response.
- Add logger.
- Allow to replace content-type header value.

6.7.11 Sdklib 1.3.x series

- Add timeout decorator.
- Add generate_url_path function.
- Add new url parameters.
- Add get, post, put, patch and delete methods.
- Add XML response parser.
- Generate docs with sphinx.

6.7.12 Sdklib 1.2.x series

- Add incognito mode.

6.7.13 Sdklib 1.1.x series

- By default, no Content-type header in requests without body or files.
- Add file attribute to sdk response.
- Allow multipart body with custom content-type in data forms.
- Allow to add custom response_class.

6.7.14 Sdklib 1.0.x series

Sdklib 1.0

- Use urllib3.

6.7.15 Sdklib 0.x series

Sdklib 0.5.2.1

- Bug fixing.

Sdklib 0.5.2

- Bug fixing.
- Allow passing files and form_parameters as tuples when request is encoded multipart

Sdklib 0.5.1

- Bug fixing.

Sdklib 0.5

- Add new parse as tuple list function.
- Add files parameter to http method.
- Infer content type header in all requests.

Sdklib 0.4.1

- Add parameters to strf timetizer functions.

Sdklib 0.4

- Add file functions.
- Add parse as tuple list function.

Sdklib 0.3

- Initial version.

Python Module Index

S

Index