ics2web Documentation

Release 0.1

bene_t valett_e pigot_a

November 29, 2014

Contents

This module is a very simple and light api for

CHAPTER 1

API

The core of the application

CHAPTER 2

IcalManage

- *IcalManage package*

2.1 api package

API Routes where you define each route and their functions.

We also use a Cache file who are invalidate every 10minutes. Feel free to change it at your ease. You can find it in the beginning of the api.py file.

```
requests_cache.install_cache('/tmp/ics-api-cache', expire_after=600)
```

1. Argument: Location of the cache directory.
2. Argument: Time before expiration in seconds.

See requests-cache doc for more information.

2.1.1 API Routes

```
@app.route('/')
```

Return "Server Running"

```
@app.route('/api/doc/')
```

Redirect to Doc ics2web.

```
@app.route('/api/get/', methods=['GET'])
def get():
    get_return_request = request.args.get('url', "")
```

Take the link provided in the URL like: "/api/get?url= <URL>" This function handle multiple error like "*Bad URL Provided*", "*Bad ICS File*", "*HTTP Exception*". It will also return a 400 : Bad request error

If everything went well, it return a json dictionnary. Else an exception is raised.

Example of json returned by the api :

```
{
  "current_events": [
    {
      "end": "2014-11-27T18:00:00",
```

```
      "name": "Test6",
      "personnes": [
        "pigot_a",
        "valett_e",
        "bene_t"
      ],
      "place": "",
      "start": "2014-11-27T16:30:00"
    },
    {
      "end": "2014-11-27T18:00:00",
      "name": "Test3",
      "personnes": [
        "pigot_a",
        "valett_e",
        "bene_t"
      ],
      "place": "",
      "start": "2014-11-27T16:30:00"
    }
  ],
  "next_day": [
    {
      "end": "2014-11-28T07:30:00",
      "name": "test_next",
      "place": "",
      "start": "2014-11-28T06:30:00"
    }
  ],
  "next_events": [
    {
      "end": "2014-11-27T20:30:00",
      "name": "test7",
      "place": "",
      "start": "2014-11-27T19:30:00"
    },
    {
      "end": "2014-11-27T21:00:00",
      "name": "test4",
      "place": "",
      "start": "2014-11-27T20:00:00"
    },
    {
      "end": "2014-11-27T22:00:00",
      "name": "Test5",
      "place": "",
      "start": "2014-11-27T21:00:00"
    },
    {
      "end": "2014-11-27T23:00:00",
      "name": "Test7",
      "place": "",
      "start": "2014-11-27T22:00:00"
    },
    {
      "end": "2014-11-27T23:00:00",
      "name": "Test8",
      "place": "",
```

```
      "start": "2014-11-27T22:00:00"
    }
  ]
}
```

2.1.2 Add Routes to api

Since the api run Flask, to add a route in the api go to api.py and add your function like this:

```
@app.route('/api/doc/')
def doc():
    # logic here
    pass
```

Which will do something when you are in the "/api/get/doc/". To be able to do some action define a function like this:

```
def doc():
```

And add your route decorator :

```
@app.route('/api/doc/')
```

Under that, put your logic in it. For exemple:

```
return redirect('http://ics2web.readthedocs.org/en/latest/#indices-and-tables')
```

Here's the complete code:

```
@app.route('/api/doc/')
def doc():
    return redirect('http://ics2web.readthedocs.org/en/latest/#indices-and-tables')
```

Explanation : Here you declare a route to "/api/get/doc". You make a function named doc() who will redirect you to Doc ics2web.

2.1.3 api module

api.api.**doc**()
Redirect to the ics2web Documentation

Returns None

api.api.**get**()
Simple method who take a ICS URL and and return a JSON object. Also handle some error.

Returns A json object of the events

api.api.**index**()
Simple message to make sure the server is running

Returns Simple string message

api.api.**read_conf**(*room=None*)
Simple method who get a ICS URL by the room ID. Easiest way to access the ICS's room by putting in the URL: /api/get/<room id>

Parameters **room** – string representing the room

Returns Json dict of events for the room

2.1.4 Subpackages

Api helpers package

api.helpers.log module

api.helpers.log.**get_status_code**(*request*)
Check the status code of the HTTP Request and log an error in the api.log file

Parameters **request** – HTTP Request

Returns Boolean

Module contents

IcalManage package

Using the parser

The parser is based on the great lib icalendar. for parsing the ICS file, but it only use the parsing part for loading an existing valid ics file.

The function is prototyped like this :

```
def ical_to_dict(stream)
```

The *stream* parameter is stream of an ical file, you can set it with a request stream, like this :

```
r = requests.get(ical_url, stream=True)
data = ical_to_dict(r)
```

Or you can simply read an local ical file

If the file content cannot be read or parsed by icalendar, the function will log an error and return *False*

The *ical_to_dict* function will return a *dict* like this :

```
{
    'current_events': list_of_events,
    'next_events': list_of_mini_events
}
```

The *current_events* contain a list containing dict with full informations about events in progress. The event in progress dict look like this :

And the *list_of_mini_events* is a list containing basics information about incoming events like this :

```
{
    "end": "2014-11-26T17:30:00",
    "name": "ert",
    "place": "Salle 301",
    "start": "2014-11-26T16:30:00"
}
```

In the actual version of the api, the parser is directly expose in JSON and returned by the *get* api function But you can use it in other place by importing it

```
from icalmanage.icalparser import ical_to_dict
```

The parser as been tested on google agenda ical file. But because of a bug when retrieving dates from the ical file, the api.icalmanage.helpers file contain a simple function for managing utc offset :

```
def set_utc(dt):
    utc_of = timezone('Europe/Paris')
    now = datetime.now()
    return dt + utc_of.utcoffset(now)
```

This function allow you to manually add the utc offset to the returned datetime object. Unfortunately the module don't has any configuration file (yet), so the timezone is hardcoded (sadly), but configuration is coming soon !

This function is call by default by the *ical_to_dict* function, which is the main function of the parser.

api.icalmanage.helpers module

api.icalmanage.helpers.**attendee_to_login**(*attendee*)
Get a list of vCalAddress and return a list of string without domain name in mail address

Parameters **attendee** – list of vCalAddress

Returns list of string

Return type list

api.icalmanage.helpers.**format_dt**(*dt*)
return a formated string from dt object It will remove utc information

Parameters **dt** (*datetime*) – datetime object to format

Returns the formated string

Return type str

api.icalmanage.helpers.**format_room**(*room*)
Format room name for display

Parameters **room** (*str*) – The string representing the room

Returns the formatted string

Return type str

api.icalmanage.helpers.**get_room**(*events*, *room*)
Get all events from events dict where event is in location room

Parameters

- **events** (*dict*) – dictionary of events
- **room** (*str*) – the name of the room

Returns a list of events for the given room

Return type list

api.icalmanage.helpers.**set_utc**(*dt*)
This function is used for correct a bug in icalendar datetime from google agenda It will add the utc offset of the localised current datetime to the dt param and return it

Parameters **dt** – the datetime to update

Returns dt

Return type datetime

api.icalmanage.icalparser module

api.icalmanage.icalparser.**check_event_current**(*ev*, *day_end*)
Check if an event is today and after now

Parameters

- **ev** – icalendar event object
- **day_end** – the end of the current day

Returns True or False for the event

api.icalmanage.icalparser.**check_event_next_day**(*ev*, *day_end*)
Check if an event occur the nex day

Parameters

- **ev** – icalendar event object
- **day_end** – the end of the current day

Returns True or False for the event

api.icalmanage.icalparser.**ev_to_partial_dict**(*ev*)
Return a dict for a given event, this dict is for next_events only

Parameters **ev** – icalendar event object

Returns a dict containing basic informations about the event

api.icalmanage.icalparser.**ical_to_dict**(*stream*)
get all event of the CURRENT day and format them to a dict ready to be encoded in json

Parameters **stream** – icalendar file object from get request

Returns a dict containing formated data

Return type dict

2.2 IcalManage package

2.2.1 Using the parser

The parser is based on the great lib icalendar. for parsing the ICS file, but it only use the parsing part for loading an existing valid ics file.

The function is prototyped like this :

```
def ical_to_dict(stream)
```

The *stream* parameter is stream of an ical file, you can set it with a request stream, like this :

```
r = requests.get(ical_url, stream=True)
data = ical_to_dict(r)
```

Or you can simply read an local ical file

If the file content cannot be read or parsed by icalendar, the function will log an error and return *False*

The *ical_to_dict* function will return a *dict* like this :

```
{
    'current_events': list_of_events,
    'next_events': list_of_mini_events
}
```

The *current_events* contain a list containing dict with full informations about events in progress. The event in progress dict look like this :

And the *list_of_mini_events* is a list containing basics information about incoming events like this :

```
{
    "end": "2014-11-26T17:30:00",
    "name": "ert",
    "place": "Salle 301",
    "start": "2014-11-26T16:30:00"
}
```

In the actual version of the api, the parser is directly expose in JSON and returned by the *get* api function But you can use it in other place by importing it

```
from icalmanage.icalparser import ical_to_dict
```

The parser as been tested on google agenda ical file. But because of a bug when retrieving dates from the ical file, the api.icalmanage.helpers file contain a simple function for managing utc offset :

```
def set_utc(dt):
    utc_of = timezone('Europe/Paris')
    now = datetime.now()
    return dt + utc_of.utcoffset(now)
```

This function allow you to manually add the utc offset to the returned datetime object. Unfortunately the module don't has any configuration file (yet), so the timezone is hardcoded (sadly), but configuration is coming soon !

This function is call by default by the *ical_to_dict* function, which is the main function of the parser.

2.2.2 api.icalmanage.helpers module

api.icalmanage.helpers.**attendee_to_login**(*attendee*)
Get a list of vCalAddress and return a list of string without domain name in mail address

Parameters **attendee** – list of vCalAddress

Returns list of string

Return type list

api.icalmanage.helpers.**format_dt**(*dt*)
return a formated string from dt object It will remove utc information

Parameters **dt** (*datetime*) – datetime object to format

Returns the formated string

Return type str

api.icalmanage.helpers.**format_room**(*room*)
Format room name for display

Parameters **room** (*str*) – The string representing the room

Returns the formatted string

Return type str

api.icalmanage.helpers.**get_room**(*events*, *room*)
Get all events from events dict where event is in location room

Parameters

- **events** (*dict*) – dictionary of events
- **room** (*str*) – the name of the room

Returns a list of events for the given room

Return type list

api.icalmanage.helpers.**set_utc**(*dt*)
This function is used for correct a bug in icalendar datetime from google agenda It will add the utc offset of the localised current datetime to the dt param and return it

Parameters **dt** – the datetime to update

Returns dt

Return type datetime

2.2.3 api.icalmanage.icalparser module

api.icalmanage.icalparser.**check_event_current**(*ev*, *day_end*)
Check if an event is today and after now

Parameters

- **ev** – icalendar event object
- **day_end** – the end of the current day

Returns True or False for the event

api.icalmanage.icalparser.**check_event_next_day**(*ev*, *day_end*)
Check if an event occur the nex day

Parameters

- **ev** – icalendar event object
- **day_end** – the end of the current day

Returns True or False for the event

api.icalmanage.icalparser.**ev_to_partial_dict**(*ev*)
Return a dict for a given event, this dict is for next_events only

Parameters **ev** – icalendar event object

Returns a dict containing basic informations about the event

api.icalmanage.icalparser.**ical_to_dict**(*stream*)
get all event of the CURRENT day and format them to a dict ready to be encoded in json

Parameters **stream** – icalendar file object from get request

Returns a dict containing formated data

Return type dict

2.3 Api helpers package

2.3.1 api.helpers.log module

api.helpers.log.**get_status_code**(*request*)
Check the status code of the HTTP Request and log an error in the api.log file

Parameters **request** – HTTP Request

Returns Boolean

2.3.2 Module contents

CHAPTER 3

Indices and tables

- *genindex*
- *modindex*
- *search*

a

A

C

D

E

F

G

I

R

S