

wol-api

1.1

Generated by Doxygen 1.8.17



# Chapter 1

## index.py, The home of our API

### 1.1 Description and Copyright notice.

This file is the main application that runs the wol-api. In this application we declare all the possible API calls that can be made with this api while it is active.

Copyright (c) 2021, 2022 Tibroness. Under the GNU General public license 2.0.

This file is part of wol-api.

This file is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <https://www.gnu.org/licenses/>.

### 1.2 Notes

- This application is still very early in its production, please handle the product at your own risk



## Chapter 2

# WOL-API

### 2.1 Watch-on-LBRY API proxy that saves data from LBRY to a database

#### Requirements:

- pipenv
- flask (pipenv install flask)
- postgresql
- libpq-dev

#### Installation instructions:

Clone the repository from Github:

Change working directory to the recently cloned folder:  
```cd wol-ap`

Install dependencies:

Fetch submodules:  
```git submodules update && cd docs/wol-docs && git pul`

Configure api settings in `/src/apimod/apcnf.py`

Run the API:

```
pipenv run python src/apimod/index.py
```

See docs for more info.

**Common issues:** P: Infinite loading after running the command.

F:

- Make sure you have configured `/src/apimod/apcnf.py` properly.



## Chapter 3

# Class Documentation

### 3.1 apimod.apcnf.Config Class Reference

#### Static Public Attributes

- string **logpath** = "~/projects/wol-api/"
- string **erppath** = "~/projects/wol-api/"
- string **postgun** = "test"
- string **postgpw** = "test"
- string **postghost** = "127.0.0.1"
- string **postgport** = "5432"
- string **postdbname** = "testdb"

#### 3.1.1 Detailed Description

This is the configuration file for the API

Here we can set the database name, username, password et cetera.

Definition at line 1 of file apcnf.py.

The documentation for this class was generated from the following file:

- apcnf.py





## Chapter 4

# File Documentation

### 4.1 index.py File Reference

The main file where the API functions are declared.

#### Functions

- def `apimod.index.ret_ok ()`  
*Returns the Documentation webpage.*
- def `apimod.index.isonline ()`  
*Returns a JSON online object.*
- def `apimod.index.error_report ()`  
*Report an error by sending a POST request to /api/report-error.*
- def `apimod.index.getlch ()`  
*Get information about a YouTube Channel.*
- def `apimod.index.getlurl ()`  
*Get information about a YouTube Video.*
- def `apimod.index.getdbcount ()`  
*Get data amount.*
- def `apimod.index.submv ()`  
*Submit a request.*

#### Variables

- `apimod.index.app = Flask(__name__)`
- `int apimod.index.debug = 0`
- `apimod.index.filename`
- `apimod.index.level`
- `apimod.index.DEBUG`
- `apimod.index.format`
- `apimod.index.con`
- `apimod.index.cursor = con.cursor()`

### 4.1.1 Detailed Description

The main file where the API functions are declared.

### 4.1.2 description\_index

This file is the main application that runs the wol-api. In this application we declare all the possible API calls that can be made with this api while it is active.

### 4.1.3 Libraries/Modules

- Flask
- os
- [apcnf.py](#)
- psycpg2
- psycpg2.extras
- json
- collections
- requests
- youtube\_dl
- datetime
- logging

### 4.1.4 ToDo

- Change the API landing page (/api/)

### 4.1.5 Author(s)

- Created by Tibroness ( <https://github.com/devbrones>) as per request by Madiator ( <https://github.com/kodxana>)
- Modified by Tibroness ( <https://github.com/devbrones>)

### 4.1.6 Function Documentation

#### 4.1.6.1 error\_report()

```
def apimod.index.error_report ( )
```

Report an error by sending a POST request to /api/report-error.

Error reporting function. TODO: This code needs to be adapted to work better with a database rather than JSON

**Parameters**

<i>error-report-type</i>	Can be "exception", "bug", or even "abcdefg"
<i>error-report-value</i>	The error message to be passed to the log, i.e. "Extension did not load."

**Returns**

204

Raises: Error in "error-reports" file.

Examples: `curl -X POST -H "Content-Type: application/json" -d '{"error-report-type":"exception","error-report-value":"Extension could not read webcontent"}'` <http://localhost:5000/api/report-error>

Writes an error in the above seen JSON format into the "error-reports" file.

Definition at line 141 of file index.py.

**4.1.6.2 getdbcount()**

```
def apimod.index.getdbcount ( )
```

Get data amount.

This function gets information on from our database, and returns a JSON Object with the number of indexed videos and channels. If no entries are found it returns 0.

**Parameters**

<i>None</i>	None
-------------	------

**Returns**

A JSON dictionary like this one: `{"channel_count":1,"video_count":2}`

Examples: `curl http://localhost:5000/api/db-count` Returns: `{"channel_count":1,"video_count":2}`

Definition at line 401 of file index.py.

**4.1.6.3 getlch()**

```
def apimod.index.getlch ( )
```

Get information about a YouTube Channel.

This function gets information on from odysee's API and saves that data to a database, and eventually returns a JSON Object. If an entry for the URL exists then it reads from that database entry.

**Parameters**

<i>url</i>	The YouTube URL
------------	-----------------

**Returns**

A JSON dictionary like this one:

Raises: Entry in log. Database functions.

Examples: curl `http://localhost:5000/api/get-lbry-channel?url=CHANNELID` Returns: `{"dtstamp": TIME_WAS_ADDED, "id": ID, "lbrych": LBRY_CHANNEL_URL, "ytch": null, "yturl": CHANNELID}`

Definition at line 171 of file index.py.

**4.1.6.4 getlurl()**

```
def apimod.index.getlurl ( )
```

Get information about a YouTube Video.

This function gets information on from odysee's API and saves that data to a database, and eventually returns a JSON Object. If an entry for the URL exists then it reads from that database entry.

**Parameters**

<i>url</i>	The YouTube URL
------------	-----------------

**Returns**

A JSON dictionary like this one: `{"dtstamp": "2022-01-04 23:26:09.752372+00:00", "id": 2, "lbrych": "@AlphaNerd#8d497e7e96c789364c56aea7a35827d2dc1eea65", "lbryurl": "@AlphaNerd#8/how-monero-works-and-why-its-a-better#a", "ytttitle": "How Monero Works (And Why its a Better Currency Than BTC)", "yturl": "QrHsFZBab4U"}`

Raises: Entry in log. Database functions.

Examples: curl `http://localhost:5000/api/get-lbry-video?url=QrHsFZBab4U` Returns: `{ "dtstamp": "2022-01-04 23:26:09.752372+00:00", "id": 2, "lbryurl": "@AlphaNerd#8/how-monero-works-and-why-its-a-better#a", "ytttitle": "How Monero Works (And Why its a Better Currency Than BTC)", "yturl": "QrHsFZBab4U" }`

Definition at line 262 of file index.py.

**4.1.6.5 isonline()**

```
def apimod.index.isonline ( )
```

Returns a JSON online object.

This feature is obsolete but I kept it anyways.

## Parameters

<i>None</i>	<i>None</i>
-------------	-------------

## Returns

[{"online":true}] Raises: 200 OK

Definition at line 126 of file index.py.

#### 4.1.6.6 ret\_ok()

```
def apimod.index.ret_ok ( )
```

Returns the Documentation webpage.

This function literally does nothing but that.

## Parameters

<i>None</i>	<i>None</i>
-------------	-------------

## Returns

The API documentation in pdf. Raises: 200 OK

Definition at line 112 of file index.py.

#### 4.1.6.7 submv()

```
def apimod.index.submv ( )
```

Submit a request.

This function lets a user submit a request to our server containing a youtube video url and a lbry url. The submissions will be manually verified by an admin.

## Parameters

<i>yturl</i>	Youtube URL
<i>lbryurl</i>	LBRY URL

**Returns**

204

Raises: Database entry.

Examples: `curl -X POST -H "Content-Type: application/json" -d '{"yturl":YTURL,"lbryurl":LBRYURL}'` <http://localhost:5000/api/submit-video> Returns: 204

Definition at line 440 of file index.py.

## 4.1.7 Variable Documentation

### 4.1.7.1 con

`apimod.index.con`

**Initial value:**

```
1 = psycopg2.connect(user=Config().postgun,  
2                       password=Config().postgpw,  
3                       host=Config().postghost,  
4                       port=Config().postgport,  
5                       database=Config().postdbname)
```

Definition at line 97 of file index.py.