aswan

Endre Márk Borza

Nov 25, 2023

CONTENTS:

1 To Setup a Remote **3**

2 Concepts **5**
2.1 Structure 5

3 Pre v1.0 laundry list **7**

4 Installation: **9**
4.1 using pip 9

5 Quickstart **11**

6 API **13**
6.1 aswan Package 13

7 Release Notes **25**
7.1 v0.0.0 25
7.2 v0.0.1 25
7.3 v0.1.0 25
7.4 v0.1.1 25
7.5 v0.2.0 25
7.6 v0.3.0 25
7.7 v0.3.1 26
7.8 v0.4.0 26
7.9 v0.4.1 26
7.10 v0.4.2 26
7.11 v0.4.3 26
7.12 v0.4.4 26
7.13 v0.4.5 26
7.14 v0.5.0 26
7.15 v0.5.1 27
7.16 v0.5.10 27
7.17 v0.5.11 27
7.18 v0.5.12 27
7.19 v0.5.13 27
7.20 v0.5.14 27
7.21 v0.5.2 27
7.22 v0.5.3 27
7.23 v0.5.4 28
7.24 v0.5.5 28
7.25 v0.5.6 28

7.26 v0.5.7 28
7.27 v0.5.8 28
7.28 v0.5.9 28

8 Indices and tables **29**

Python Module Index **31**

Index **33**

collect and organize data into a T1 data depot named after the Aswan Dam

Collect and compress data from the internet for later parsing

- quick, parallel, customizable to collect
- compressed to store
- quick to sync with a remote store
  - sync to continue collecting
  - sync to parse
- immutable collection

CHAPTER

ONE

TO SETUP A REMOTE

set the environment variables `ASWAN_AUTH_HEX` and `ASWAN_AUTH_PASS` according to the zimmauth package, and `ASWAN_REMOTE` with the name of the default remote.

CHAPTER

TWO

CONCEPTS

- objects
  - saved by collection events
- events
  - collection
  - registration (v2: registration for parsing)
  - (v2) parsing
- runs
  - manual run vs automated run
    - makes manual adding of urls easy but revertible
  - has unique id
  - generates events
  - linked to a specific version of the code
    - ideally commit hash + pip freeze
- statuses
  - determined by base status + runs integrated
  - contains
    - what urls need to be collected
    - (v2) what collected objects need to be parsed
  - sqlite file, constantly trimmed

2.1 Structure

- objects
  - 00, 01, …
- runs
  - run-hash
    - context.yaml
      - commit-hash, pip-freeze, …

        * events.zip
- statuses
    - status-hash
        * context.yaml
            · parent-status, integrated
        * db.sqlite.zip
- current-run
    - context.yaml
    - events
        * these to be compressed into ../runs
    - status.sqlite
- there is a 'TEST' status
    - cannot be integrated whatever is based on it
    - a test run can be made on it…

when starting a run:

- check if current-run is empty
    - if not, fail with
- find latest status
    - if it has not integrated all past runs, create a new status that has
- start collection (+ registration)
- either stops or breaks, all events and objects are saved to disk
- if properly stops, move and compress stuff
    - based on one that was the starter, and current run id

CHAPTER

THREE

PRE V1.0 LAUNDRY LIST

- parallelize push / pull
- parsing/connection/broken session error docs
- transferring / ignoring cookies
- template projects
  - oddsportal
    - updating thingy, based on latest match in season
  - footy
  - rotten
  - boxoffice

CHAPTER

FOUR

INSTALLATION:

4.1 using pip

```
pip install aswan
```

CHAPTER

FIVE

QUICKSTART

```
from aswan import __version__
```

CHAPTER

SIX

API

6.1 aswan Package

Data collection manager

6.1.1 Functions

add_url_params(url, params)	
get_json(url[, params])	
get_soup(url[, params, browser, headless, ...])	
run_simple_project(urls_for_handlers, name)	

add_url_params

aswan.**add_url_params**(*url*, *params*)

get_json

aswan.**get_json**(*url: str*, *params: dict | None = None*)

get_soup

aswan.**get_soup**(*url: str*, *params: dict | None = None*, *browser=False*, *headless=True*, *headers=None*)

run_simple_project

aswan.**run_simple_project**(*urls_for_handlers: Dict[*RequestHandler | BrowserHandler*, Iterable[str]]*, *name: str*, *sync=False*, *remote: str | bool | None = None*)

6.1.2 Classes

AswanDepot(name[, local_root])	
BrokenSessionError	
BrowserHandler()	
BrowserJsonHandler()	
BrowserSoupHandler()	
ConnectionError	
ConnectionSession(depot_path[, is_browser, ...])	
ObjectStore(root[, hash_fun, compression, ...])	class for storing and retrieving objects downloaded
ParsedCollectionEvent(cev, store)	
Project(name[, local_root, distributed_api, ...])	
ProxyAuth(user, password)	
ProxyBase()	
RequestHandler()	
RequestJsonHandler()	
RequestSoupHandler()	
Statuses()	
WebExtHandler()	

AswanDepot

class aswan.**AswanDepot**(*name: str*, *local_root: Path | None = None*)

Bases: RemoteMixin

BrokenSessionError

exception aswan.**BrokenSessionError**

BrowserHandler

class aswan.**BrowserHandler**

Bases: UrlHandlerBase

Attributes Summary

eager	
headless	

Methods Summary

handle_driver(driver)	runs after get.
is_session_broken(result)	exception when getting source
parse(source)	
start_session(browser)	start session if browser is needed

Attributes Documentation

eager: bool = False

headless: bool = False

Methods Documentation

handle_driver(*driver: Chrome*)

runs after get. if returns something, that is forwarded to parse

is_session_broken(*result: Exception*)

exception when getting source

if error occurs during handle browser, it comes here

parse(*source*)

start_session(*browser: Chrome*)

start session if browser is needed

BrowserJsonHandler

class aswan.**BrowserJsonHandler**

Bases: _JsonMixin, *BrowserHandler*

BrowserSoupHandler

class aswan.**BrowserSoupHandler**

Bases: _SoupMixin, *BrowserHandler*

ConnectionError

exception aswan.**ConnectionError**

ConnectionSession

class aswan.**ConnectionSession**(*depot_path: ~pathlib.Path | None = None*, *is_browser=False*, *headless=True*, *eager=False*, *proxy_cls: type[~aswan.security.ProxyBase] = <class 'aswan.security.NoProxy'>*, *is_webext: bool = False*)

Bases: ActorBase

Methods Summary

consume(task)	
get_parsed_response(url[, handler, params])	
proc_result(task, out, status)	
stop()	if any cleanup needed

Methods Documentation

consume(*task: HandlingTask*)

get_parsed_response(*url*, *handler=<aswan.url_handler.RequestSoupHandler object>*, *params: dict | None = None*)

proc_result(*task: HandlingTask*, *out: Any*, *status: str*)

stop()

if any cleanup needed

ObjectStore

class aswan.**ObjectStore**(*root*, *hash_fun=<built-in function openssl_sha3_512>*, *prefix_chars: int = 2*, *compression=8*, *timeout=60*)

Bases: `object`

class for storing and retrieving objects downloaded

Parameters
root – object store root

Methods Summary

dump(obj)
dump_bytes(buf[, ext])
dump_json(obj)
dump_pickle(obj)
dump_str(s[, ext])
purge([clear_dirs])
read(name)
read_bytes(name)
read_json(name)
read_pickle(name)
read_str(name)

Methods Documentation

dump(*obj: list | dict | str | bytes*)

dump_bytes(*buf: bytes*, *ext=None*) → str

dump_json(*obj: list | dict*) → str

dump_pickle(*obj*) → str

dump_str(*s: str*, *ext=None*) → str

purge(*clear_dirs=True*)

read(*name: str*)

read_bytes(*name: str*) → bytes

read_json(*name: str*) → list | dict

read_pickle(*name: str*)

read_str(*name: str*) → str

ParsedCollectionEvent

class aswan.**ParsedCollectionEvent**(*cev: CollEvent*, *store:* ObjectStore)

Bases: object

Attributes Summary

content
url

Attributes Documentation

content

url

Project

class aswan.**Project**(*name: str*, *local_root: str | None = None*, *distributed_api='mp'*, *max_displays: int = 4*, *max_cpu_use: float = 2.0*, *batch_multiplier=16*, *debug=False*)

Bases: object

Attributes Summary

resource_limits

Methods Summary

commit_current_run()	
continue_run([inprogress, parsing_error, ...])	
register_handler(handler)	
register_module(mod)	
run([urls_to_register, urls_to_overwrite, ...])	run project

Attributes Documentation

resource_limits

Methods Documentation

commit_current_run()

continue_run(*inprogress=True*, *parsing_error=False*, *conn_error=False*, *sess_broken=False*, *force_sync=False*, *urls_to_register: Dict[Type[*RequestHandler | BrowserHandler*], Iterable[str]] | None = None*, *urls_to_overwrite: Dict[Type[*RequestHandler | BrowserHandler*], Iterable[str]] | None = None*, *keep_running=True*)

register_handler(*handler: Type[*RequestHandler | BrowserHandler*]*)

register_module(*mod*)

run(*urls_to_register: Dict[Type[*RequestHandler | BrowserHandler*], Iterable[str]] | None = None*, *urls_to_overwrite: Dict[Type[*RequestHandler | BrowserHandler*], Iterable[str]] | None = None*, *test_run=False*, *keep_running=True*, *force_sync=False*)

run project

test runs on a basic local thread

ProxyAuth

class aswan.**ProxyAuth**(*user: str*, *password: str*)

Bases: object

ProxyBase

class aswan.**ProxyBase**

Bases: object

Attributes Summary

expiration_secs
max_at_once
port_no
prefix

Methods Summary

get_chrome_options()	
get_creds()	if authentication is needed
get_requests_dict()	
set_new_host()	

Attributes Documentation

expiration_secs = 31536000

max_at_once = None

port_no = 80

prefix = 'http'

Methods Documentation

get_chrome_options()

get_creds() → *ProxyAuth* | None
if authentication is needed

get_requests_dict()

set_new_host()

RequestHandler

class aswan.**RequestHandler**
Bases: UrlHandlerBase

Methods Summary

handle_driver(session)	runs before get.
is_session_broken(result)	either response code, or exception
parse(blob)	
start_session(session)	starts session if no browser needed

Methods Documentation

handle_driver(*session: Session*)
runs before get. can set/update cookies/headers

is_session_broken(*result: int | Exception*)
either response code, or exception

parse(*blob*)

start_session(*session: Session*)
starts session if no browser needed

RequestJsonHandler

class aswan.**RequestJsonHandler**
Bases: _JsonMixin, *RequestHandler*

RequestSoupHandler

class aswan.**RequestSoupHandler**
Bases: _SoupMixin, *RequestHandler*

Statuses

class aswan.**Statuses**
Bases: object

Attributes Summary

CACHE_LOADED
CONNECTION_ERROR
PARSING_ERROR
PERSISTENT_CACHED
PERSISTENT_PROCESSED
PROCESSED
PROCESSING
SESSION_BROKEN
TODO

Attributes Documentation

CACHE_LOADED = 'CL'

CONNECTION_ERROR = 'CE'

PARSING_ERROR = 'PE'

PERSISTENT_CACHED = 'PC'

PERSISTENT_PROCESSED = 'PP'

PROCESSED = 'D'

PROCESSING = 'processing'

SESSION_BROKEN = 'SB'

TODO = 'todo'

WebExtHandler

class aswan.**WebExtHandler**

Bases: UrlHandlerBase

Attributes Summary

max_in_parallel
max_retries
restart_session_after
wait_time

Methods Summary

handle_driver(app)
is_session_broken(result)
parse(we_resp)
start_session(_)

Attributes Documentation

max_in_parallel: int | None = 1

max_retries: int = 5

restart_session_after: int = inf

wait_time = 1

Methods Documentation

handle_driver(*app*)

is_session_broken(*result: int | Exception*)

parse(*we_resp: bytes*)

start_session(*_*)

6.1.3 Class Inheritance Diagram

CHAPTER

SEVEN

RELEASE NOTES

7.1 v0.0.0

- first release of aswan, yay!!

7.2 v0.0.1

- revert to 3.8 for ray

7.3 v0.1.0

integrate atqo

7.4 v0.1.1

rm ray from basic

7.5 v0.2.0

major simplification

7.6 v0.3.0

for the devs

7.7 v0.3.1

even better

7.8 v0.4.0

limit parallelism and some efficiency

7.9 v0.4.1

improve response handling

7.10 v0.4.2

align with dz

7.11 v0.4.3

silence

7.12 v0.4.4

silence

7.13 v0.4.5

hope

7.14 v0.5.0

adapt to atqo

7.15 v0.5.1

fix parallel processing

7.16 v0.5.10

logging and slight remote fixes

7.17 v0.5.11

dependency fix

7.18 v0.5.12

dependency fix

7.19 v0.5.13

add brotli req

7.20 v0.5.14

webext handler

7.21 v0.5.2

expose monitor better

7.22 v0.5.3

monitor improvement, more parallelization

7.23 v0.5.4

rm recursion

7.24 v0.5.5

caching in depot

7.25 v0.5.6

cant download to Path

7.26 v0.5.7

abs path

7.27 v0.5.8

post_status kwarg and error handling

7.28 v0.5.9

better pulling

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

a
aswan, 13

INDEX

A

add_url_params() (*in module aswan*), 13
aswan
    module, 13
AswanDepot (*class in aswan*), 15

B

BrokenSessionError, 15
BrowserHandler (*class in aswan*), 15
BrowserJsonHandler (*class in aswan*), 16
BrowserSoupHandler (*class in aswan*), 16

C

CACHE_LOADED (*aswan.Statuses attribute*), 22
commit_current_run() (*aswan.Project method*), 19
CONNECTION_ERROR (*aswan.Statuses attribute*), 22
ConnectionError, 16
ConnectionSession (*class in aswan*), 16
consume() (*aswan.ConnectionSession method*), 16
content (*aswan.ParsedCollectionEvent attribute*), 18
continue_run() (*aswan.Project method*), 19

D

dump() (*aswan.ObjectStore method*), 17
dump_bytes() (*aswan.ObjectStore method*), 17
dump_json() (*aswan.ObjectStore method*), 17
dump_pickle() (*aswan.ObjectStore method*), 17
dump_str() (*aswan.ObjectStore method*), 17

E

eager (*aswan.BrowserHandler attribute*), 15
expiration_secs (*aswan.ProxyBase attribute*), 20

G

get_chrome_options() (*aswan.ProxyBase method*), 20
get_creds() (*aswan.ProxyBase method*), 20
get_json() (*in module aswan*), 13
get_parsed_response() (*aswan.ConnectionSession method*), 16
get_requests_dict() (*aswan.ProxyBase method*), 20
get_soup() (*in module aswan*), 13

H

handle_driver() (*aswan.BrowserHandler method*), 15
handle_driver() (*aswan.RequestHandler method*), 21
handle_driver() (*aswan.WebExtHandler method*), 23
headless (*aswan.BrowserHandler attribute*), 15

I

is_session_broken() (*aswan.BrowserHandler method*), 15
is_session_broken() (*aswan.RequestHandler method*), 21
is_session_broken() (*aswan.WebExtHandler method*), 23

M

max_at_once (*aswan.ProxyBase attribute*), 20
max_in_parallel (*aswan.WebExtHandler attribute*), 23
max_retries (*aswan.WebExtHandler attribute*), 23
module
    aswan, 13

O

ObjectStore (*class in aswan*), 17

P

parse() (*aswan.BrowserHandler method*), 15
parse() (*aswan.RequestHandler method*), 21
parse() (*aswan.WebExtHandler method*), 23
ParsedCollectionEvent (*class in aswan*), 18
PARSING_ERROR (*aswan.Statuses attribute*), 22
PERSISTENT_CACHED (*aswan.Statuses attribute*), 22
PERSISTENT_PROCESSED (*aswan.Statuses attribute*), 22
port_no (*aswan.ProxyBase attribute*), 20
prefix (*aswan.ProxyBase attribute*), 20
proc_result() (*aswan.ConnectionSession method*), 16
PROCESSED (*aswan.Statuses attribute*), 22
PROCESSING (*aswan.Statuses attribute*), 22
Project (*class in aswan*), 18
ProxyAuth (*class in aswan*), 19
ProxyBase (*class in aswan*), 19
purge() (*aswan.ObjectStore method*), 17

R

read() (*aswan.ObjectStore method*), 17
read_bytes() (*aswan.ObjectStore method*), 17
read_json() (*aswan.ObjectStore method*), 17
read_pickle() (*aswan.ObjectStore method*), 18
read_str() (*aswan.ObjectStore method*), 18
register_handler() (*aswan.Project method*), 19
register_module() (*aswan.Project method*), 19
RequestHandler (*class in aswan*), 20
RequestJsonHandler (*class in aswan*), 21
RequestSoupHandler (*class in aswan*), 21
resource_limits (*aswan.Project attribute*), 19
restart_session_after (*aswan.WebExtHandler attribute*), 23
run() (*aswan.Project method*), 19
run_simple_project() (*in module aswan*), 14

S

SESSION_BROKEN (*aswan.Statuses attribute*), 22
set_new_host() (*aswan.ProxyBase method*), 20
start_session() (*aswan.BrowserHandler method*), 15
start_session() (*aswan.RequestHandler method*), 21
start_session() (*aswan.WebExtHandler method*), 23
Statuses (*class in aswan*), 21
stop() (*aswan.ConnectionSession method*), 16

T

TODO (*aswan.Statuses attribute*), 22

U

url (*aswan.ParsedCollectionEvent attribute*), 18

W

wait_time (*aswan.WebExtHandler attribute*), 23
WebExtHandler (*class in aswan*), 22