

---

**aswan**

**Endre Márk Borza**

**Dec 04, 2023**



## CONTENTS:

<b>1</b>	<b>To Setup a Remote</b>	<b>3</b>
<b>2</b>	<b>Concepts</b>	<b>5</b>
2.1	Structure . . . . .	5
<b>3</b>	<b>Pre v1.0 laundry list</b>	<b>7</b>
<b>4</b>	<b>Installation:</b>	<b>9</b>
4.1	using pip . . . . .	9
<b>5</b>	<b>Quickstart</b>	<b>11</b>
<b>6</b>	<b>API</b>	<b>13</b>
6.1	aswan Package . . . . .	13
<b>7</b>	<b>Release Notes</b>	<b>25</b>
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
<b>8</b>	<b>Indices and tables</b>	<b>29</b>
	<b>Python Module Index</b>	<b>31</b>
	<b>Index</b>	<b>33</b>

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



## **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.





## 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

## 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



## INSTALLATION:

### 4.1 using pip

```
pip install aswan
```



## QUICKSTART

```
from aswan import __version__
```





## 6.1 aswan Package

Data collection manager

### 6.1.1 Functions

---

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

---

#### **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

<i>eager</i>
<i>headless</i>

### Methods Summary

<i>handle_driver</i> (driver)	runs after get.
<i>is_session_broken</i> (result)	exception when getting source
<i>parse</i> (source)	
<i>start_session</i> (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

<code>consume(task)</code>	
<code>get_parsed_response(url[, handler, params])</code>	
<code>proc_result(task, out, status)</code>	
<code>stop()</code>	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

<code>content</code>
<code>url</code>

#### 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

<code>resource_limits</code>
------------------------------

#### Methods Summary

<code>commit_current_run()</code>	
<code>continue_run([inprogress, parsing_error, ...])</code>	
<code>register_handler(handler)</code>	
<code>register_module(mod)</code>	
<code>run([urls_to_register, urls_to_overwrite, ...])</code>	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

<i>get_chrome_options()</i>	
<i>get_creds()</i>	if authentication is needed
<i>get_requests_dict()</i>	
<i>set_new_host()</i>	

## 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

<i>handle_driver</i> (session)	runs before get.
<i>is_session_broken</i> (result)	either response code, or exception
<i>parse</i> (blob)	
<i>start_session</i> (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

<a href="#">CACHE_LOADED</a>
<a href="#">CONNECTION_ERROR</a>
<a href="#">PARSING_ERROR</a>
<a href="#">PERSISTENT_CACHED</a>
<a href="#">PERSISTENT_PROCESSED</a>
<a href="#">PROCESSED</a>
<a href="#">PROCESSING</a>
<a href="#">SESSION_BROKEN</a>
<a href="#">TODO</a>

### 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

<i>max_in_parallel</i>
<i>max_retries</i>
<i>restart_session_after</i>
<i>wait_time</i>

### Methods Summary

<i>handle_driver</i> (app)
<i>is_session_broken</i> (result)
<i>parse</i> (we_resp)
<i>start_session</i> (_)

### 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



## 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



## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`



## PYTHON MODULE INDEX

### a

aswan, [13](#)



## 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