
PyFunceble Documentation

Release 3.1.20

Nissar Chababy (@funilrys)

Jun 15, 2020

Contents

1	What can PyFuncible do?	3
2	Contributors	5
3	Special Thanks	7
4	Supporting the project	9
4.1	GitHub Sponsor	9
4.2	Ko-Fi	9
5	License	11
6	Code of Conduct	13
6.1	Our Pledge	13
6.2	Our Standards	13
6.3	Our Responsibilities	14
6.4	Scope	14
6.5	Enforcement	14
6.6	Attribution	14
7	They use-d PyFuncible!	15
8	Frequently Asked Questions	17
8.1	How to speed up the test process?	17
8.1.1	I have multiple CPU	17
8.1.2	I do not have multiple CPU	18
9	Known issues	19
10	Discovered an issue?	21
11	Need help?	23
12	Need information?	25
13	Dead-Hosts	27
13.1	Why Dead-Hosts?	27
13.2	How do Dead-Hosts work?	27

13.3	History of Dead-Hosts	27
14	Installation	29
14.1	Requirements	29
14.1.1	Python 3.6.9+	29
14.1.2	colorama	29
14.1.3	cryptography	30
14.1.4	dnspython	30
14.1.5	domain2idna	30
14.1.6	PyMySQL	30
14.1.7	python-box	30
14.1.8	python-dotenv	30
14.1.9	PyYAML	30
14.1.10	requests	30
14.1.11	setuptools	30
14.1.12	urllib3	30
14.2	Stable version	31
14.2.1	Using pip	31
14.2.2	Using the AUR (for Arch Linux users)	31
14.2.3	Using docker (hub)	32
14.2.4	Using conda	32
14.2.5	Pure Python method	32
14.3	Development version	32
14.3.1	For development	32
14.3.2	For usage	33
14.4	First steps	35
15	Update	37
15.1	Stable version	37
15.1.1	Using pip	37
15.1.2	Using the AUR (for Arch Linux users)	38
15.1.3	Using docker (hub)	38
15.1.4	Using conda	38
15.1.5	Pure Python method	38
15.2	Development version	39
15.2.1	For development	39
15.2.2	For usage	39
16	Configuration	41
16.1	Location	41
16.1.1	Problematics	41
16.1.2	Repository clone	41
16.1.3	Travis CI	41
16.1.4	GitLab CI/CD	42
16.1.5	Linux and MacOS (Darwin Kernel)	42
16.1.6	Windows	42
16.1.7	Custom location	43
16.2	Autoconfiguration	43
16.3	Indexes	43
16.3.1	adblock	43
16.3.2	aggressive	44
16.3.3	auto_continue	44
16.3.4	command	44
16.3.5	command_before_end	44

16.3.6	cooldown_time	44
16.3.7	custom_ip	45
16.3.8	days_between_db_retest	45
16.3.9	days_between_inactive_db_clean	45
16.3.10	db_type	45
16.3.11	debug	46
16.3.12	dns_lookup_over_tcp	46
16.3.13	dns_server	46
16.3.14	filter	47
16.3.15	generate_complements	47
16.3.16	generate_hosts	47
16.3.17	generate_json	47
16.3.18	header_printed	47
16.3.19	hierarchical_sorting	48
16.3.20	iana_whois_server	48
16.3.21	idna_conversion	48
16.3.22	inactive_database	48
16.3.23	less	48
16.3.24	local	49
16.3.25	logs	49
16.3.26	maximal_processes	49
16.3.27	mining	49
16.3.28	multiprocess	49
16.3.29	multiprocess_merging_mode	49
16.3.30	no_files	50
16.3.31	no_special	50
16.3.32	no_whois	50
16.3.33	plain_list_domain	50
16.3.34	print_dots	50
16.3.35	quiet	51
16.3.36	referer	51
16.3.37	reputation	51
16.3.38	shadow_file	51
16.3.39	share_logs	51
16.3.40	show_execution_time	52
16.3.41	show_percentage	52
16.3.42	simple	52
16.3.43	split	52
16.3.44	syntax	52
16.3.45	timeout	53
16.3.46	ci	53
16.3.47	ci_autosave_commit	53
16.3.48	ci_autosave_final_commit	53
16.3.49	ci_autosave_minutes	53
16.3.50	ci_distribution_branch	53
16.3.51	ci_branch	54
16.3.52	unified	54
16.3.53	use_reputation_data	54
16.3.54	verify_ssl_certificate	54
16.3.55	whois_database	55
16.3.56	wildcard	55
16.3.57	user_agent	55
16.3.58	outputs	56
16.3.59	status	64

16.3.60	http_codes	66
16.3.61	links	68
17	Usage	71
17.1	From a terminal	71
17.1.1	--help	71
17.1.2	-v --version	71
17.1.3	Source	71
17.1.4	Source filtering, decoding, conversion and expansion	73
17.1.5	Test control	73
17.1.6	DNS (resolver) control	76
17.1.7	Databases	76
17.1.8	Output control	78
17.1.9	Multiprocessing	80
17.1.10	CI / CD	81
17.1.11	Unique actions	83
17.1.12	Global overview	84
17.2	From a GitLab CI/CD environment	88
17.2.1	Configuration	88
17.3	From a Travis CI container	89
17.3.1	Configuration	89
17.3.2	Getting a GitHub token	91
17.3.3	Encrypting the token for future usage under the Travis CIs' containers	91
17.4	Using the PyFuncible (Python) API	91
17.4.1	Get the availability of domains or IP	91
17.4.2	Get the availability of URL	93
17.4.3	Complete dataset while getting the availability of domains, IPs or URL	94
17.4.4	Set custom configuration index while getting the availability of domains, IPs or URL	94
17.4.5	Globally	94
17.4.6	Check the syntax of domains	95
17.4.7	Check the syntax of IPv4s	96
17.4.8	Check the syntax of URLs	97
18	Responses	99
18.1	Subject	99
18.2	Status	99
18.2.1	ACTIVE	99
18.2.2	INACTIVE	100
18.2.3	INVALID	100
18.2.4	VALID	100
18.2.5	MALICIOUS	100
18.2.6	SANE	100
18.3	Source	101
18.3.1	HTTP Code	101
18.3.2	SYNTAX	101
18.3.3	DNSLOOKUP	101
18.3.4	SPECIAL	101
18.4	_status (API)	102
18.5	_status_source (API)	103
18.6	dns_lookup (API)	103
18.6.1	Format 1: Normal DNS Lookup	103
18.6.2	Format 2: get_hosts_by_addr (local subjects?)	103
18.7	HTTP Code	103
18.7.1	As active	104

18.7.2	As potentially active	104
18.7.3	As inactive or potentially inactive	105
18.8	Expiration Date	106
18.8.1	Unknown (CLI)	106
18.8.2	None (API)	106
18.8.3	A date	106
18.9	whois_record (API)	106
18.10	whois_server (API)	106
18.11	domain_syntax_validation (API)	107
18.12	subdomain_syntax_validation (API)	107
18.13	url_syntax_validation (API)	107
18.14	ipv4_syntax_validation (API)	107
18.15	ipv6_syntax_validation (API)	107
18.16	ipv4_range_syntax_validation (API)	107
18.17	ipv6_range_syntax_validation (API)	108
19	API Documentation	109
19.1	Endpoints	109
19.2	File generation while using the API	115
19.2.1	Global configuration load	115
19.2.2	Local configuration load	115
20	Components	117
20.1	AdBlock/Filter list decoding	117
20.1.1	Why do we need it?	117
20.1.2	How does it work?	117
20.1.3	How to use it?	117
20.2	Autocontinue	118
20.2.1	Why do we need it?	118
20.2.2	How does it work?	118
20.2.3	How to use it?	118
20.3	Autosave	118
20.3.1	Why do we need it?	118
20.3.2	How does it work?	119
20.3.3	How to use it?	119
20.4	Certificate verification	119
20.4.1	Why do we need it?	119
20.4.2	How does it work?	119
20.4.3	How to use it?	119
20.5	Cleaning	120
20.5.1	Why do we need it?	120
20.5.2	How does it work?	120
20.5.3	How to clean?	120
20.5.4	Differences between simple and complete cleaning?	120
20.6	Complements Generation	120
20.6.1	Why do we need it?	121
20.6.2	How does it work?	121
20.6.3	How to use it?	121
20.7	Configuration	121
20.7.1	Why do we need it?	121
20.7.2	How does it work?	121
20.7.3	How to configure?	122
20.8	Custom DNS Server	122
20.8.1	Why do we need it?	122

20.8.2	How does it work?	122
20.8.3	How to use it?	122
20.9	Custom User-Agent	122
20.9.1	Why do we need it?	122
20.9.2	How does it work?	123
20.9.3	How to use it?	123
20.9.4	What if we don't give a custom User-Agent?	123
20.10	Databases	124
20.10.1	Why do we use "databases"?	124
20.10.2	How do we manage them?	124
20.10.3	Warnings around Database (self) management	124
20.10.4	Databases types	124
20.10.5	Known limitations with MySQL and MariaDB	125
20.10.6	Search string	126
20.11	Directory Structure	126
20.11.1	Why do we need it?	126
20.11.2	How does it work?	126
20.11.3	How to generate it manually?	126
20.12	DNS Lookup	126
20.12.1	Why do we need it?	126
20.12.2	How does it work?	127
20.13	Environment variables	127
20.13.1	Dotenv files	127
20.13.2	What do we use and why ?	128
20.14	Execution time	128
20.14.1	Why do we need it?	128
20.14.2	How does it work?	128
20.14.3	How to use it?	129
20.15	List filtering	129
20.15.1	Why do we need it?	129
20.15.2	How does it work?	129
20.15.3	How to use it?	129
20.16	IANA Root Zone Database	129
20.16.1	Why do we need it?	129
20.16.2	How does it work?	130
20.16.3	How to generate it manually?	130
20.17	Test in/for local hostnames, IPs, components	130
20.17.1	Why do we need it?	130
20.17.2	How does it work?	130
20.17.3	How to use it?	130
20.18	Logs Sharing	131
20.18.1	Why do we need it?	131
20.18.2	What do we share/collect?	131
20.18.3	How to share logs?	131
20.19	Mining	131
20.19.1	Why do we need it?	131
20.19.2	How does it work?	132
20.19.3	How to use it?	132
20.20	Multiprocessing	132
20.20.1	Why do we need it?	132
20.20.2	How does it work?	133
20.20.3	How to use it?	133
20.21	Outputed Files	134
20.21.1	Why do we need it?	134

20.21.2	How does it work?	134
20.22	Percentage	135
20.22.1	Why do we need it?	135
20.22.2	How does it work?	135
20.22.3	How to use it?	135
20.23	The Public Suffix List	135
20.23.1	Why do we need it?	135
20.23.2	How does it work?	136
20.23.3	How to generate it manually?	136
20.24	Sorting	136
20.24.1	Why do we need it?	136
20.24.2	How does it work?	136
20.24.3	How to activate the hierarchical sorting?	137
20.25	Whois Lookup	137
20.25.1	Why do we need it?	137
20.25.2	How does it work?	137
20.25.3	How to use it?	138
21	Contribute	139
21.1	Submit changes	139
21.2	Coding conventions	139
21.2.1	Format	139
21.2.2	Linting	139
21.2.3	Coverage	140
21.2.4	Distribution	140
21.2.5	Commit	140
21.2.6	Git Hook	140
22	Code Documentation	143
22.1	Logic representation	143
22.1.1	Domain testing	145
22.1.2	IP testing	146
22.1.3	URL testing	148
22.2	Check	149
22.2.1	Problematic	149
22.2.2	Documentation	149
22.3	Exceptions	151
22.3.1	Problematic	151
22.3.2	Documentation	151
22.4	Abstracts	153
22.4.1	Problematic	153
22.4.2	Documentation	153
22.5	CLI	156
22.5.1	Problematic	156
22.5.2	Documentation	157
22.6	Config	160
22.6.1	Problematic	160
22.6.2	Documentation	160
22.7	Converter	162
22.7.1	Problematic	162
22.7.2	Documentation	162
22.8	Core	165
22.8.1	Problematic	165
22.8.2	Documentation	165

22.9	Database	170
22.9.1	Problematic	170
22.9.2	Documentation	170
22.10	Downloader	173
22.10.1	Problematic	173
22.10.2	Documentation	173
22.11	Engine	175
22.11.1	Problematic	175
22.11.2	Documentation	175
22.12	Extractor	181
22.12.1	Problematic	181
22.12.2	Documentation	181
22.13	Helpers	182
22.13.1	Problematic	182
22.13.2	Documentation	182
22.14	Lookup	190
22.14.1	Problematic	190
22.14.2	Documentation	190
22.15	Output	199
22.15.1	Problematic	199
22.15.2	Documentation	200
22.16	Status	206
22.16.1	Problematic	206
22.16.2	Documentation	206
23	Indices and tables	213
	Python Module Index	215
	Index	217



build failing

coverage 100%

license Apache-2.0

pypi v3.1.0

issues 10 open

downloads 479/day

downloads 3.4k/week

downloads 16k/month

PyFunceble is the little sister of [Funceble](#) which was archived on 13th March 2018.

Its main objective is to provide the availability of domains, IPs and since recently URL by generating an accurate result based on results from WHOIS, NSLOOKUP and HTTP status codes.

PyFunceble provides some useful features for continuous testing.

As an example, its auto-continue system coupled with its auto-save system allows it to run nice and smoothly under Travis CI without even reaching Travis CI time restriction. In the other side, its internal inactive database system let INACTIVE and INVALID caught domains, IPs or URLs being automatically retested over time on next run.

CHAPTER 1

What can PyFunceble do?

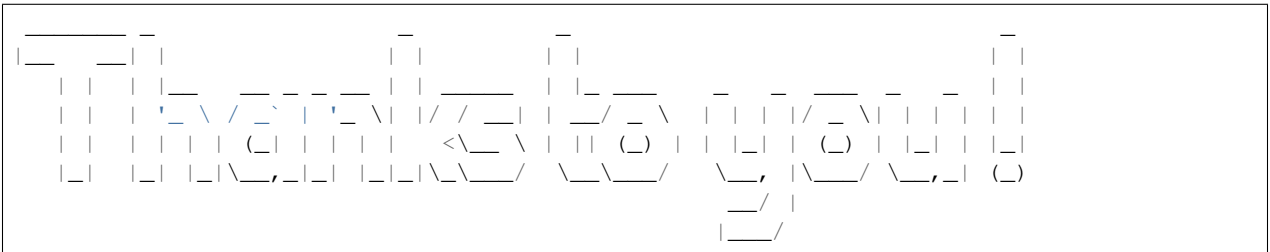
- Test the availability of a domain.
- Test the availability of an IPv4.
- Test the availability of an IPv6.
- Test the availability of a URL.
- Test the availability of a domain/DNS name in a private or local network.
- Test the availability of an IPv4 in a private or local network.
- Test the availability of an IPv6 in a private or local network.
- Test the availability of a URL in a private or local network.
- Test the syntax of a domain.
- Test the syntax of an IPv4.
- Test the syntax of an IPv6.
- Test the syntax of a URL.
- Test the AlienVault's reputation of an IPv4.
- Test of domain or IP which are present into an Adblock formatted file.
- Test from a given raw link.
- Test using multiprocessing (from CLI only).
- Save test result(s) in file(s) (hosts file, plain text and/or JSON format).
- Save test result in a SQL database.
- Show test result(s) on screen.
- Show percentage of each status (ACTIVE, INACTIVE, INVALID)
- Sort outputs hierarchically.
- "Mining" of domain or IP which are related to the tested element.

- Auto-continuation of tests in case of system crash or script stop.
- Filtering of a file content.
 - This feature will let us for example test all blogspot domain of the given file no matter the content of the file.
- Set the user-agent to use for the tests.
- Give some analytic depending of the HTTP status code (ACTIVE, POTENTIALLY_ACTIVE, POTENTIALLY_INACTIVE, SUSPICIOUS).
- Retest overtime of INACTIVE and INVALID domains.
- Print the execution time on screen and file.
- Customisation of the different option via command-line arguments or configuration file.
- Continuous tests under Travis CI or GitLab CI/CI
 - ... with the help of an auto saving and database system.
 - Set the branch to push the result to. For the autosaving system.
 - Set the minimal time before we autosave in order to avoid CI/CD limitation.
 - Set a command to execute at the end of the test.
 - Set the commit message for the autosaving system.
- ... and a lot more!

CHAPTER 2

Contributors

Thanks to those awesome peoples for their awesome and crazy idea(s), contribution(s) and or issue report which made or make [PyFunceble](#) a better tool.



- Daniel - [@dnmTX](#)
- hawkeye116477 - [@hawkeye116477](#)
- Imre Kristoffer Eilertsen - [@DandelionSprout](#)
- jawz101 - [@jawz101](#)
- kowith337 - [@kowith337](#)
- Mitchell Krog - [@mitchellkrogza](#)
- NeolithEra - [@NeolithEra](#)
- Odysseus - [@Odysseus](#)
- opav - [@opav](#)
- Reza Rizqullah - [@ybreza](#)
- ScriptTiger - [@ScriptTiger](#)
- sjhgvr - [@sjhgvr](#)
- speedmann - [@speedmann](#)
- spirillen - [@spirillen](#)

- The Unknown - [@AnonymousPoster](#)
- WaLLy3K - [@WaLLy3K](#)
- xxcriticxx - [@xxcriticxx](#)
- ZeroDot1 - [@ZeroDot1](#)

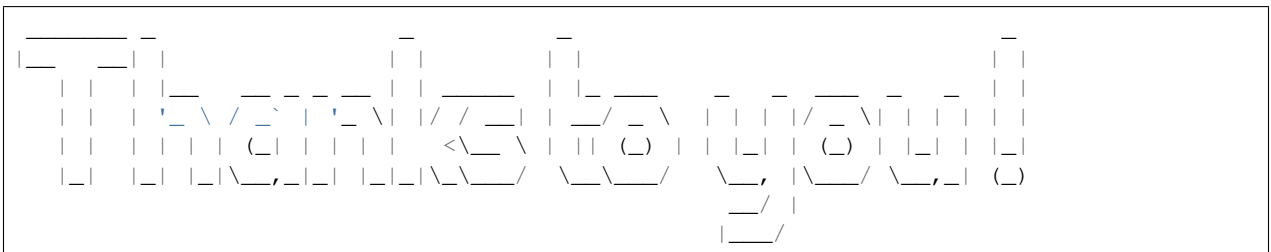
CHAPTER 3

Special Thanks

Thanks to those awesome organization(s), tool(s) and or people(s) for

- Their awesome documentation
- Their awesome repository
- Their awesome tool/software/source code
- Their breaking reports
- Their contributions
- Their current work/purpose
- Their promotion of Py-Funceble
- Their support
- Their testings reports

which helped and/or still help me build, test and or make [PyFunceble](#) a better tool.



- Adam Warner - [@PromoFaux](#)
- Adblock Plus - [@adblockplus](#)
- asciinema - [@asciinema](#)
- Bob Halley - [@rthalley](#) (DNSPython)
- Chris Griffith - [@cdgriffith](#) (Box)

- Daniel - @dnmTX
- Jonathan Hartley - @tartley (colorama)
- IANA - ICANN (Root Zone Database)
- Kenneth Reitz - @kennethreitz (requests)
- Mitchell Krog - @mitchellkrogza
- Mohammad Fares - @faressoft (Terminalizer)
- Pi-Hole - @pi-hole
- Public Suffix List - @publicsuffix
- Reza Rizqullah - @ybreza
- Saurabh Kumar - @theskumar (python-dotenv)
- ScriptTiger - @ScriptTiger
- SMed79 - @SMed79
- spirillen - @spirillen
- The YAML Project - @yaml (pyyaml)
- yWorks - (yEd Graph Editor)

Supporting the project

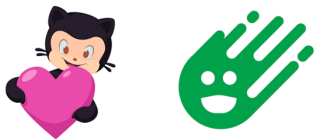
PyFunceble, Dead-Hosts, and all other analogue projects are powered by my free time and a lot of coffee!

If you like this project and/or it have helped you. Please consider buying me more coffee.

4.1 GitHub Sponsor

@funilrys is part of the GitHub Sponsor program!

GitHub will match all donation for the coming months!



Sponsor me!

4.2 Ko-Fi

Don't want to use the GitHub Sponsor program ? Single donation are welcome too!



Buy me a coffee!

CHAPTER 5

License

Copyright 2017, 2018, 2019, 2020 Nissar Chababy

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

6.1 Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, gender identity and expression, level of experience, nationality, personal appearance, race, religion, or sexual identity and orientation.

6.2 Our Standards

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or electronic address, without explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

6.3 Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

6.4 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

6.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team at contact@funilrys.com. The project team will review and investigate all complaints, and will respond in a way that it deems appropriate to the circumstances. The project team is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

Project maintainers who do not follow or enforce the Code of Conduct in good faith may face temporary or permanent repercussions as determined by other members of the project's leadership.

6.6 Attribution

This Code of Conduct is adapted from the [Contributor Covenant](http://contributor-covenant.org/version/1/4), version 1.4, available at <http://contributor-covenant.org/version/1/4>.

CHAPTER 7

They use-d PyFunceble!

Just a list of tool/repository/code/infrastructure which uses or used PyFunceble!

- [.pl-host-file](#)
- [add.2o7Net](#)
- [add.Dead](#)
- [add.Risk](#)
- [add.Spam](#)
- [adfilt](#)
- [Asads-Hosts](#)
- [Badd-Boyz-Hosts](#)
- [Badger Sett](#)
- [blacklist](#)
- [CoinBlockerLists](#)
- [cookies_filters](#)
- [DNS test](#)
- [Facebook Zero Hosts Block](#)
- [finnish-easylist-addition](#)
- [Frellwits-filter-lists](#)
- [Google AdService and ID.Google tracking hosts](#)
- [KADhosts](#)
- [MobileAdTrackers](#)
- [My Privacy DNS](#)
- [Phishing-URL-Testing-Database-of-Link-Statuses](#)

- [Phishing.Database](#)
- [polish-adblock-filters](#)
- [polish-pihole-filters](#)
- [pornhosts](#)
- [Stop.Google.Analytics.Ghost.Spam.HOWTO](#)
- [The-Big-List-of-Hacked-Malware-Web-Sites](#)
- [Top-Attacking-IP-Addresses-Against-Wordpress-Sites](#)
- [Ultimate.Hosts.Blacklist // Ultimate-Hosts-Blacklist](#)
- [UncheckyAds](#)

Frequently Asked Questions

8.1 How to speed up the test process?

Warning: Beware, when talking about speed a lot of things have to be taken in consideration. Indeed here is a non exhaustive list of things which fluctuate testing speed.

- Bandwidth.
- DNS Server response time.
- CPU.
- ISP blocking a big amount of connection to the outside world.
- Our databases management (do not apply for MySQL and MariaDB format).
- Amount of data to test.
- ...

8.1.1 I have multiple CPU

Simply use the `-m | --multiprocess` argument to activate the usage of multiple processes. You should in addition to the `-m` specify the `-p | --processes` argument.

If `-p | --processes` is avoided, the script will use the number of available CPU cores.

You might therefore wish to specify the number of simultaneous processes to be used, otherwise you will be “unable” to use the computer/server for other things while running PyFuncible as all of your CPU threads is used by PyFuncible.

Note: A good number for `-p` is your number of `CPU_cores - 1`, to leave room for other processes to work. Unless you have a dedicated installation for this work.

Inside a Unix based system, you can use this code snippet to see how many CPU and cores you have.

```
$ lscpu | grep -E '^Thread|^Core|^Socket|^CPU\('
```

or

```
$ ``$(nproc --ignore=1)``
```

This will count the number of CPU threads subtracted 1 to use for DB, SQL. If you runs PyFunceble on your workstation you might subtract 2 threads, or you computer will be pretty "dead"

See also ``man nproc`` or ``nproc --help``

Warning: DO NOT try to exceed your total number of CPU cores (as `-p` | `--processes`), if you want to keep your machine somehow alive and healthy.

8.1.2 I do not have multiple CPU

In case you only have a single core, you should disable the usage of the WHOIS lookup by adding the `--no-whois` to your command line or switching the value of `no_whois` to `True` in your configuration file.

As example:

```
$ PyFunceble --no-whois -d example.net
```

This action will speed up the script because it only use the equivalent of `nslookup` and the `HTTP status code` to determine the availability status.

Warning: PyFunceble request the WHOIS record in order to avoid specific false positive cases. If the usage of WHOIS request is disabled, all domains which are still registered but not assigned to an IP address, would be flagged as `INACTIVE`.

It's not a problem if you keep/use the database system because the domain will be retested over time. But please keep in mind, that without the database system, the accuracy of the result is not guaranteed.

CHAPTER 9

Known issues

This is the list of issues which are not or will not be fixed (yet...).

- Under Travis CI the coloration may not be shown.
- Under GitLab CI/CD the coloration may not be shown.
- When using either Oracle's MySQL or MariaDB-Server you will have to be either SUPER-PRIVILEGED user (root) or be able to do `set global log_bin_trust_function_creators=1`; Read more about this in the [components/databases](#)

CHAPTER 10

Discovered an issue?

Want to talk with [@funilrys](#) about it? Contact [@funilrys](#) on Keybase.

Want to make it public? Fill out a [new issue](#).

Don't want to fill out an issue on GitHub? Let's discuss it on our [issues team](#) on Keybase.

CHAPTER 11

Need help?

Do you need help with the usage of PyFuncible? We can help!
Want to talk with [@funilrys](#) about it? Contact [@funilrys](#) on Keybase.
Otherwise, let's discuss it on our [support team](#) on Keybase!

CHAPTER 12

Need information?

You can get information and discuss about PyFuncible on our [main Keybase team](#). @Funilrys (me) will do his best to keep it up-to-date with some information about the development and other news about PyFuncible.

If you need more information, the [support team](#) or [issues team](#), is the place to ask!

13.1 Why Dead-Hosts?

We are conscient that PyFunceble may run for days. We would therefore offer you, to have your list. [Request for Dead-Hosts](#) testing.

13.2 How do Dead-Hosts work?

[Dead-Hosts](#) is like a wrapper of PyFunceble. Indeed, we use Travis CI containers in order to test lists with PyFunceble.

Once a list is set up inside our infrastructure, PyFunceble will test the list regularly and the Dead-Hosts infrastructure will produce a `clean.list` file which represents the list of domains/IP/URL which remains or became `ACTIVE`.

In addition, if needed, we can set up a list of GitHub username to `@ping` once a new test is finished.

Do you want your list to be tested at [Dead-Hosts](#) ? You can [request it](#) !

13.3 History of Dead-Hosts

The project started on 3rd March 2017 at [funilrys/dead-hosts](#).

The original idea was to test [Funceble](#) against hosts file in order to find bugs inside [Funceble](#) but also letting me (@funilrys) have a shorter hosts file.

On 23rd January 2018, I (@funilrys) decided to shut [funilrys/dead-hosts](#) down, as it became impossible to test all members of the project without having to wait for weeks. I then decided it was time to move on to another level.

That was the beginning of [Dead-Hosts](#).

Today [Dead-Hosts](#)'s is objective to provide project and lists to maintainers or individuals, with the help of PyFunceble, to provide more information about their favourite project/list or domains, IP or URL.

14.1 Requirements

Here is the list of requirements:

- Python 3.6.9+
- colorama
- cryptography
- dnspython
- domain2idna
- PyMySQL
- python-box
- python-dotenv
- PyYAML
- requests
- setuptools
- urllib3

14.1.1 Python 3.6.9+

The specification 3.6.9+ is because we test PyFuncible (daily) in all version from 3.6.9. Otherwise, more semantically, PyFuncible is written for all Python 3.6+ version.

14.1.2 colorama

As we use some coloration coloration, colorama is required.

14.1.3 cryptography

As we use some cryptography algorithm for message hashing as example, `cryptography` is required.

14.1.4 dnspython

As we use it to do the DNS lookups, `dnspython` is required.

14.1.5 domain2idna

As we propose the conversion of domains to IDNA, `domain2idna` is required.

Note: `domain2idna` is maintained and developed by [Nissar Chababy \(@funilrys\)](#), the developer of PyFunceble. Its source code can be found [on GitHub](#).

14.1.6 PyMySQL

As we propose the MariaDB or MySQL database types, `PyMySQL` is required.

14.1.7 python-box

As we use `python-box` for a better code access to the configuration, it is required.

14.1.8 python-dotenv

As we are able to load dotenv files, `python-dotenv` is required.

14.1.9 PyYAML

As our configuration file is written in `.yaml`, `PyYAML` is required.

14.1.10 requests

As we use `requests` multiple times to communicate with webservices, `requests` is required.

14.1.11 setuptools

As we use `install_requires=xx` inside our `setup.py`, `setuptools` is required.

14.1.12 urllib3

You should normally already have it. But as we handle some of its errors while using `requests`, `urllib3` is required.

14.2 Stable version

14.2.1 Using pip

From PyPi

```
$ pip3 install --user PyFunceble
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFunceble into containers like - for example - Travis CI.

From GitHub

```
$ pip3 install --user git+https://github.com/funilrys/PyFunceble.git@master  
↪ #egg=PyFunceble
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFunceble into containers like - for example - Travis CI.

14.2.2 Using the AUR (for Arch Linux users)

The package can be found at <https://aur.archlinux.org/packages/pyfunceble/>.

With makepkg

```
$ wget https://aur.archlinux.org/cgit/aur.git/plain/PKGBUILD?h=pyfunceble  
$ makepkg  
$ sudo pacman -U pyfunceble*.tar.xz
```

With your favorite AUR helper

Warning: We do not recommend any AUR helper but keep in mind that some AUR helpers are “better” than other. For more information about your current (or any other) AUR helper please report to [the ArchWiki page](#).

```
$ yourFavoriteAurHelper -S pyfuncible
```

14.2.3 Using docker (hub)

The image description can be found at <https://hub.docker.com/r/pyfuncible/pyfuncible>

```
$ docker pull pyfuncible/pyfuncible
```

14.2.4 Using conda

Our repository is located at <https://anaconda.org/pyfuncible/pyfuncible>

```
conda install -c conda-forge -c pyfuncible pyfuncible=3
```

14.2.5 Pure Python method

Execute the following and enjoy PyFuncible!

```
$ git clone https://github.com/funilrys/PyFuncible.git
$ cd PyFuncible
$ python3 setup.py test
$ python3 setup.py install # Avoid this if you want to uninstall one day.
$ pip3 install --user -e .
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFuncible into containers like - for example - Travis CI.

14.3 Development version

The development version of PyFuncible represents the `dev` branch. It's intended for the development of next features but is always at a usable state.

Indeed, We should not push to the `dev` branch until we are sure that the new commit does not break or introduce critical issue under PyFuncible.

14.3.1 For development

Execute the following and let's hack PyFuncible!

Note: We highly recommend you to develop PyFuncible under a `virtualenv`.

```
$ git clone https://github.com/funilrys/PyFunceble.git
$ cd PyFunceble
$ git checkout dev
$ virtualenv venv
$ source venv/bin/activate
$ pip3 install -e .
```

Note: After installing with:

```
$ source venv/bin/activate
$ pip3 install -e .
```

- you only need to update the repository.
 - you don't have to rerun the `pip` command.
-

14.3.2 For usage

Using `pip`

Execute one of the following and enjoy PyFunceble!

From PyPi

```
$ pip3 install --user PyFunceble-dev
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFunceble into containers like - for example - Travis CI.

From GitHub

```
$ pip3 install --user git+https://github.com/funilrys/PyFunceble.git@dev
↪ #egg=PyFunceble
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFunceble into containers like - for example - Travis CI.

Using the AUR (for Arch Linux users)

The package can be found at <https://aur.archlinux.org/packages/pyfunceble-dev/>.

With makepkg

```
$ wget https://aur.archlinux.org/cgit/aur.git/plain/PKGBUILD?h=pyfunceble-dev
$ makepkg
$ sudo pacman -U pyfunceble-dev*.tar.xz
```

With your favorite AUR helper

Warning: We do not recommend any AUR helper but keep in mind that some AUR helpers are “better” than other. For more information about your current (or any other) AUR helper please report to [the ArchWiki page](#).

```
$ yourFavoriteAurHelper -S pyfunceble-dev
```

Using docker (hub)

The image description can be found at <https://hub.docker.com/r/pyfunceble/pyfunceble-dev>

```
$ docker pull pyfunceble/pyfunceble-dev
```

Using conda

Our repository is located at <https://anaconda.org/pyfunceble/pyfunceble-dev>

```
conda install -c conda-forge -c pyfunceble pyfunceble-dev=3
```

Pure Python method

Execute the following and enjoy PyFunceble!

```
$ git clone https://github.com/funilrys/PyFunceble.git
$ cd PyFunceble && git checkout dev
$ python3 setup.py test
$ python3 setup.py install # Avoid this if you want to uninstall one day.
$ pip3 install --user -e .
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFunceble into containers like - for example - Travis CI.

14.4 First steps

Make sure that you can run

```
$ PyFunceble --version
```

and enjoy PyFunceble!!

15.1 Stable version

15.1.1 Using pip

From PyPi

```
$ pip3 install --user --upgrade PyFunceble
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFunceble into containers like - for example - Travis CI.

From GitHub

```
$ pip3 install --user --upgrade git+https://github.com/funilrys/PyFunceble.git@master  
↪ #egg=PyFunceble
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFunceble into containers like - for example - Travis CI.

15.1.2 Using the AUR (for Arch Linux users)

With makepkg

```
$ wget https://aur.archlinux.org/cgit/aur.git/plain/PKGBUILD?h=python-pyfunceble
$ makepkg
$ sudo pacman -U python-pyfunceble*.tar.xz
```

With your favorite AUR helper

Warning: We do not recommend any AUR helper but keep in mind that some AUR helpers are “better” than other. For more information about your current (or any other) AUR helper please report to [the ArchWiki page](#).

```
$ yourFavoriteAurHelper -Syu python-pyfunceble
```

15.1.3 Using docker (hub)

The image description can be found at <https://hub.docker.com/r/pyfunceble/pyfunceble>

```
$ docker pull pyfunceble/pyfunceble
```

15.1.4 Using conda

Our repository is located at <https://anaconda.org/pyfunceble/pyfunceble>

```
conda update -c conda-forge -c pyfunceble pyfunceble
```

15.1.5 Pure Python method

Execute the following and enjoy PyFunceble!

```
$ cd PyFunceble
$ git checkout master && git fetch origin && git merge origin/master
$ python3 setup.py test
$ python3 setup.py install # Avoid this if you want to uninstall one day.
$ pip3 install --user --upgrade -e .
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFunceble into containers like - for example - Travis CI.

15.2 Development version

15.2.1 For development

```
$ cd PyFunceble && git checkout dev
$ git fetch origin && git merge origin/dev
```

Note: As you previously installed with

```
$ . venv/bin/activate && pip3 install -e .
```

Only code/repository update is required.

15.2.2 For usage

Using pip

From PyPi

```
$ pip3 install --user --upgrade PyFunceble-dev
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFunceble into containers like - for example - Travis CI.

From GitHub

```
$ pip3 install --user --upgrade git+https://github.com/funilrys/PyFunceble.git@dev
↪ #egg=PyFunceble
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFunceble into containers like - for example - Travis CI.

Using the AUR (for Arch Linux users)

With makepkg

```
$ wget https://aur.archlinux.org/cgit/aur.git/plain/PKGBUILD?h=python-pyfuncible-dev
$ makepkg
$ sudo pacman -U python-pyfuncible-dev*.tar.xz
```

With your favorite AUR helper

Warning: We do not recommend any AUR helper but keep in mind that some AUR helpers are “better” than other. For more information about your current (or any other) AUR helper please report to [the ArchWiki page](#).

```
$ yourFavoriteAurHelper -Syu python-pyfuncible-dev
```

Using docker (hub)

The image description can be found at <https://hub.docker.com/r/pyfuncible/pyfuncible-dev>

```
$ docker pull pyfuncible/pyfuncible-dev
```

Using conda

Our repository is located at <https://anaconda.org/pyfuncible/pyfuncible-dev>

```
conda update -c conda-forge -c pyfuncible pyfuncible-dev
```

Pure Python method

Execute the following and enjoy PyFuncible!

```
$ cd PyFuncible && git checkout dev
$ git fetch origin && git merge origin/dev
$ python3 setup.py test
$ python3 setup.py install # Avoid this if you want to uninstall one day.
$ pip3 install --user --upgrade -e .
```

Note: We recommend the `--user` flag which installs the required dependencies at the user level. More information about it can be found on [pip documentation](#).

Warning: We do not recommend the `--user` flag when using PyFuncible into containers like - for example - Travis CI.

16.1 Location

16.1.1 Problematics

- How can we create a more efficient way to work with configuration?
- How can we make the configuration file(s) available globally so that PyFuncible can be run everywhere in the user workspace?

To answer those problematics, we moved the configuration location elsewhere in the place where most users expect to have their configuration file(s).

16.1.2 Repository clone

If you cloned the repository and you're trying to test from a cloned directory (the one with for example `CONTRIBUTING.md`) we consider the configuration directory as the current one.

Note: This behavior allows us to not modify the way we develop PyFuncible.

16.1.3 Travis CI

Under [Travis CI](#), we search or initiate the configuration at the directory we are currently located.

Warning: We don't care about the distribution, as long as the `TRAVIS_BUILD_DIR` environment variable is set, we search or initiate the configuration in the current directory.

Note: If you want to force the directory where we should work, you can initiate the `PYFUNCEBLE_CONFIG_DIR` environment variable with the path where we should work.

16.1.4 GitLab CI/CD

Under [GitLab CI/CD](#), we search or initiate the configuration at the directory we are currently located.

Warning: We don't care about the distribution, as long as the `PROJECT_CI` and `GITLAB_CI` environment variables are set, we search or initiate the configuration in the current directory.

Note: If you want to force the directory where we should work, you can initiate the `PYFUNCEBLE_CONFIG_DIR` environment variable with the path where we should work.

16.1.5 Linux and MacOS (Darwin Kernel)

Under Linux and MacOS, we look for the following directories in their order. If any configuration directory is found, the system proposes you to install them automatically on the first configuration file.

1. `~/ .config/PyFunceble`
2. `~/ .PyFunceble`
3. `${PWD}`

Note: If the parent directory does not exist, we move to the next possible location in the given order.

This means that under most Linux distributions and MacOS versions, we consider `~/ .config/PyFunceble` as the configuration location. But if the `~/ .config` directory does not exist, we fallback to `~/ .PyFunceble` as the configuration location.

16.1.6 Windows

As mentioned by [Pat Altimore's Blog](#), we used the Per user configuration files synchronized across domain joined machines via Active Directory Roaming section in order to understand what we should do to find our configuration directory.

Under Windows, we look for the following directories in their order. If any configuration directory is found, the system proposes you to install them automatically on the first configuration file.

1. `%APPDATA%\PyFunceble` (environnement variable)
2. `%CD%`

Note: `%CD%` is explained by the `set` command (`set /?`):

`%CD%` - expands to the current directory string.

Note: If the parent directory does not exist, we move to the next possible location in the given order.

This means that under most Windows versions, we consider %APPDATA%\PyFunceble - also known as C:\Users\userName\AppData\Roaming\PyFunceble- as the configuration location. But if the %APPDATA% directory does not exist, we fall back to the current directory as the configuration location.

16.1.7 Custom location

Sometimes, you may find yourself in a position where you absolutely do not want PyFunceble to use its default configuration location.

For that reason, if you set your desired configuration location along with the PYFUNCEBLE_CONFIG_DIR environment variable, we take that location as the (default) configuration location.

16.2 Autoconfiguration

Sometimes, you may find yourself in a position that you do not or you can't answer the question which asks you if you would like to install the default configuration file.

For that reason, if you set PYFUNCEBLE_AUTO_CONFIGURATION as an environment variable with what you want an assignment, we do not ask that question. We simply do what we have to do without asking anything.

16.3 Indexes

This page will try to detail each configuration available into .PyFunceble.yaml along with the location of where we are looking for the configuration file.

16.3.1 adblock

Type: boolean

Default value: False

Description: Enable / disable the adblock format decoding.

Note: If this index is set to True, every time we read a given file, we try to extract the elements that are present.

We basically only decode the adblock format.

Note: If this index is set to False, every time we read a given file, we will consider one line as an element to test.

16.3.2 aggressive

Type: boolean

Default value: False

Description: Enable / disable some aggressive settings.

Warning: This option is available but please keep in mind that the some settings which it enable are experimental.

16.3.3 auto_continue

Type: boolean

Default value: True

Description: Enable / disable the auto continue system.

16.3.4 command

Type: string

Default value: ""

Description: Set the command to run before each commit (except the final one).

Note: The parsed command is called only if `auto_continue` and `ci` are set to `True`.

16.3.5 command_before_end

Type: string

Default value: ""

Description: Set the command to run before the final commit.

Note: The parsed command is called only if `auto_continue` and `ci` are set to `True`.

Note: Understand by final commit the commit which will deliver the last element we have to test.

16.3.6 cooldow_time

Type:: float

Default value: null

Description: Set the cooldown time to apply between each test.

Note: This index take only effect from the CLI. Not from the API.

16.3.7 custom_ip

Type: string

Default value: "0.0.0.0"

Description: Set the custom IP to use when we generate a line in the hosts file format.

Note: This index has no effect if `generate_hosts` is set to `False`.

16.3.8 days_between_db_retest

Type: integer

Default value: 1

Description: Set the number of day(s) between each retest of the `INACTIVE` and `INVALID` elements which are present into `inactive_db.json`.

Note: This index has no effect if `inactive_database` is set to `False`.

16.3.9 days_between_inactive_db_clean

Type: integer

Default value: 28

Description: Set the numbers of days since the introduction of a subject into `inactive-db.json` for it to qualifies for deletion.

Note: This index has no effect if `inactive_database` is set to `False`.

16.3.10 db_type

Type: string

Default value: json

Available values: json, mariadb, mysql

Description: Set the database type to use everytime we create a database.

Note: This feature is applied to the following subsystems:

- Autocontinue physically located (JSON) at `output/continue.json`.
- InactiveDB physically located (JSON) at `[config_dir]/inactive_db.json`.

- Mining physically located (JSON) at `[config_dir]/mining.json`.
 - WhoisDB physically located (JSON) at `[config_dir]/whois.json`.
-

16.3.11 debug

Type: `boolean`

Default value: `False`

Description: Enable / disable the generation of debug file(s).

Note: This index has no effect if `logs` is set to `False`

Warning: Do not touch this index unless you have good reason to.

Warning: Do not touch this index unless you have been invited to.

16.3.12 dns_lookup_over_tcp

Type: `boolean`

Default value: `False`

Description: Make all DNS lookup with TCP instead of UDP.

16.3.13 dns_server

Type: `None` or `list`

Default value: `null`

Description: Set the DNS server(s) to work with.

Note: When a list is given the following format is expected.

```
dns_server:
- dns1.example.org
- dns2.example.org
```

Note: You can specify a port number to use to the DNS server if needed.

As example:

```
- 127.0.1.53:5353
```


Warning: We expect a DNS server(s). If no DNS server(s) is given. You'll almost for certain get all results as INACTIVE

This could happens in case you use `--dns -f`

16.3.14 filter

Type: string

Default value: ""

Description: Set the element to filter.

Note: This index should be initiated with a regular expression.

16.3.15 generate_complements

Type: boolean

Default value: False

Description: Enable / disable the generation and test of the complements.

Note: A complement is for example `example.org` if `www.example.org` is given and vice-versa.

16.3.16 generate_hosts

Type: boolean

Default value: True

Description: Enable / disable the generation of the hosts file(s).

16.3.17 generate_json

Type: boolean

Default value: False

Description: Enable / disable the generation of the JSON file(s).

16.3.18 header_printed

Type: boolean

Default value: False

Description: Say to the system if the header has been already printed or not.

Warning: Do not touch this index unless you have a good reason to.

16.3.19 hierarchical_sorting

Type: boolean

Default value: False

Description: Say to the system if we have to sort the list and the outputs in a hierarchical order.

16.3.20 iana_whois_server

Type: string

Default value: whois.iana.org

Description: Set the server to call to get the whois referer of a given element.

Note: This index is only used when generating the `iana-domains-db.json` file.

Warning: Do not touch this index unless you a have good reason to.

16.3.21 idna_conversion

Type: boolean

Default value: False

Description: Tell the system to convert all domains to IDNA before testing.

Note: We use `domain2idna` for the conversion.

16.3.22 inactive_database

Type: boolean

Default value: True

Description: Enable / Disable the usage of a database to store the INACTIVE and INVALID element to retest overtime.

16.3.23 less

Type: boolean

Default value: True

Description: Enable / Disable the output of every information of screen.

16.3.24 local

Type: boolean

Default value: False

Description: Enable / Disable the execution of the test(s) in a local or private network.

16.3.25 logs

Type: boolean

Default value: True

Description: Enable / Disable the output of all logs.

16.3.26 maximal_processes

Type: integer

Default value: 25

Description: Set the number of maximal simultaneous processes to use/create/run.

Warning: If you omit the `--processes` argument, we overwrite the default with the number of available CPU.

16.3.27 mining

Type: boolean

Default value: True

Description: Enable / Disable the mining subsystem.

16.3.28 multiprocess

Type: boolean

Default value: False

Description: Enable / Disable the usage of multiple processes instead of the default single process.

16.3.29 multiprocess_merging_mode

Type: string

Default value: end

Available values: end, live

Description: Set the multiprocess merging mode.

Note: With the `end` value, the merging of cross process data is made at the very end of the current instance.

Note: With the `live` value, the merging of cross process data is made after the processing of the maximal number of process.

Which means that if you allow 5 processes, we will run 5 tests, merge, run 5 tests, merge and so on until the end.

16.3.30 `no_files`

Type: `boolean`

Default value: `False`

Description: Enable / Disable the generation of any file(s).

16.3.31 `no_special`

Type: `boolean`

Default value: `False`

Description: Enable / Disable the usage of the SPECIAL rules - which are describes in the source column section.

16.3.32 `no_whois`

Type: `boolean`

Default value: `False`

Description: Enable / Disable the usage of `whois` in the tests.

16.3.33 `plain_list_domain`

Type: `boolean`

Default value: `False`

Description: Enable / Disable the generation of the plain list of elements sorted by statuses.

Warning: Do not touch this index unless you a have good reason to.
--

16.3.34 `print_dots`

Type: `boolean`

Default value: `False`

Description: Enable / Disable the printing of dots (.) for the skipped subjects.

Note: The idea is to not given the impression that we hang. Because most of the time, we don't.

16.3.35 quiet

Type: boolean

Default value: False

Description: Enable / Disable the generation of output on the screen.

16.3.36 referer

Type: string

Default value: " "

Description: Set the referer of the element that is currently under test.

Warning: Do not touch this index unless you a have good reason to.

16.3.37 reputation

Type: boolean

Default value: False

Description: Enable / disable the reputation (only) testing.

Warning: If this index is set to `True`, we **ONLY** check for reputation, not availability nor syntax.

16.3.38 shadow_file

Type: boolean

Default value: False

Description: Enable / Disable the usage and generation of a shadow file before a the test of a file.

Note: The shadow file, will just contain the actual list of subjects to test.

16.3.39 share_logs

Type: boolean

Default value: True

Description: Enable / disable the logs sharing.

Note: This index has no effect if `logs` is set to `False`.

16.3.40 `show_execution_time`

Type: `boolean`

Default value: `False`

Description: Enable / disable the output of the execution time.

16.3.41 `show_percentage`

Type: `boolean`

Default value: `True`

Description: Enable / disable the output of the percentage of each status.

16.3.42 `simple`

Type: `boolean`

Default value: `False`

Description: Enable / disable the simple output mode.

Note: If this index is set to `True`, the system will only return the result in format: `tested.element STATUS`.

16.3.43 `split`

Type: `boolean`

Default value: `True`

Description: Enable / disable the split of the results files.

Note: Understand with “results files” the mirror of what is shown on screen.

16.3.44 `syntax`

Type: `boolean`

Default value: `False`

Description: Enable / disable the syntax (only) testing.

Warning: If this index is set to `True`, we **ONLY** check for syntax, not availability nor reputation.

16.3.45 timeout

Type: integer

Default value: 5

Description: Set the timeout to apply everytime it's possible to set one.

16.3.46 ci

Type: boolean

Default value: False

Description: Enable / disable the CI autosaving system.

Warning: Do not activate this index unless you are using PyFunceble under a supported CI environment/platform.

16.3.47 ci_autosave_commit

Type: string

Default value: "PyFunceble - AutoSave"

Description: Set the default commit message we want to use when have to commit (save) but our tests are not yet completed.

16.3.48 ci_autosave_final_commit

Type: string

Default value: "PyFunceble - Results"

Description: Set the default final commit message we want to use when we all tests are finished.

16.3.49 ci_autosave_minutes

Type: integer

Default value: 15

Description: Set the minimum of minutes we have to run before to automatically save our test results.

Note: As many services are setting a rate limit per IP, it's a good idea to set this value between 1 and 15 minutes.

16.3.50 ci_distribution_branch

Type: string

Default value: master

Description: Set the git branch where we are going to push our results.

Note: The difference between this and `ci_branch` is the fact that this branch will get the result only when the test were finished under the given `ci_branch`.

As example, this allow us to have 2 branches:

- `processing` (ci branch), for the tests with PyFunceble.
 - `master` (ci distribution branch), for the distribution of the results of PyFunceble.
-

16.3.51 `ci_branch`

Type: `string`

Default value: `master`

Description: Set the git branch where we are going to push our results.

16.3.52 `unified`

Type: `boolean`

Default value: `False`

Description: Enable / Disable the generation of the unified results.

Note: This index has no effect if `split` is set to `True`.

16.3.53 `use_reputation_data`

Type: `boolean`

Default value: `False`

Description: Enable / Disable the usage of reputation data while testing the availability of a given subject.

Warning: This only have an effect when used along with the availability test.

16.3.54 `verify_ssl_certificate`

Type: `boolean`

Default value: `False`

Description: Enable / Disable the verification of the SSL/TLS certificate when testing for URL.

Warning: If you set this index to `True`, you may get **false positive** result.

Indeed if the certificate is not registered to the CA or is simply invalid and the domain is still alive, you will always get `INACTIVE` as output.

16.3.55 whois_database

Type: boolean

Default value: True

Description: Enable / Disable the usage of the whois database to avoid/bypass whois server requests rate limit.

16.3.56 wildcard

Type: boolean

Default value: False

Description: Enable / Disable the test of wildcards when testing for syntax.

Warning: This is not taken into consideration if `syntax` is set to `False`.

16.3.57 user_agent

Type: dict

Description: Configures the user agent.

`user_agent[browser]`

Type: string

Default value: chrome

Description: Sets the browser to get the latest user agent from. Available values: chrome, edge, firefox, ie, opera, safari

Warning: This option is not taken in consideration if `user_agent[custom]` is not set to null.

`user_agent[platform]`

Type: string

Default value: linux

Description: Sets the platform to get the latest user agent for. Available values: linux, macosx, win10

Warning: This option is not taken in consideration if `user_agent[custom]` is not set to null.

`user_agent [custom]`

Type: string

Default value: null

Description: Sets the user agent to use.

Warning: Setting this index will overwrite the choices made into `user_agent[platform]` and `user_agent[browser]`.

16.3.58 outputs

Type: dict

Description: Set the needed output tree/names.

Warning: If you choose to change anything please consider deleting our `output/` directory and the `dir_structure*.json` files.

`outputs[default_files]`

Type: dict

Description: Set the default name of some important files.

`outputs[default_files][dir_structure]`

Type: string

Default value: `dir_structure.json`

Description: Set the default filename of the file which has the structure to re-construct.

Note: This index has no influence with `dir_structure_production.json`

`outputs[default_files][iana]`

Type: string

Default value: `iana-domains-db.json`

Description: Set the default filename of the file which has the formatted copy of the IANA root zone database.

`outputs[default_files][inactive_db]`

Type: string

Default value: inactive_db.json

Description: Set the default filename of the file which will save the list of elements to retest overtime.

`outputs[default_files][results]`

Type: string

Default value: results.txt

Description: Set the default filename of the file which will save the formatted copy of the public suffix database.

`outputs[default_files][public_suffix]`

Type: string

Default value: public-suffix.json

Description: Set the default filename of the file which will save the mirror of what is shown on screen.

`outputs[default_files][mining]`

Type: string

Default value: mining.json

Description: Set the default filename of the file which will save the temporary list of mined subject to test.

`outputs[default_files][whois_db]`

Type: string

Default value: whois_db.json

Description: Set the default filename of the file which will save the whois information for caching.

`outputs[db_type]`

Type: dict

Description: Set the default name of some important files related to the db_type index.

`outputs[db_type][directory]`

Type: string

Default value: `.db_type/`

Description: Set the default directory where we have to save the SQL files which we are later going to be used for database generation/creation.

`outputs[db_type][files]`

Type: dict

Description: Set the default name of some important files related to the `db_type` index.

`outputs[domains]`

Type: dict

Description: Set the default name of some important files related to the `plain_list_domain` index.

`outputs[domains][directory]`

Type: string

Default value: `domains/`

Description: Set the default directory where we have to save the plain list of elements for each status.

`outputs[domains][filename]`

Type: string

Default value: `list`

Description: Set the default filename of the file which will save the plain list of elements.

`outputs[hosts]`

Type: dict

Description: Set the default name of some important files related to the `generate_hosts` index.

`outputs[hosts][directory]`

Type: string

Default value: `hosts/`

Description: Set the default directory where we have to save the hosts files of the elements for each status.

outputs[hosts][filename]

Type: string

Default value: hosts

Description: Set the default filename of the file which will save the hosts files of the elements.

outputs[json]

Type: dict

Description: Set the default name of some important files related to the `generate_json` index.

outputs[json][directory]

Type: string

Default value: json

Description: Set the default directory where we have to save the JSON files of the elements for each status.

outputs[json][filename]

Type: string

Default value: dump.json

Description: Set the default filename of the file which will save the JSON files of the elements.

outputs[complements]

Type: dict

Description: Set the default name of some important files/directories related to the `generate_complements` index.

outputs[complements][directory]

Type: string

Default value: complements

Description: Set the default directory where we have to save the complements related files sorted by status.

outputs[analytic]

Type: dict

Description: Set the default name of some important files and directories related to the `generate_hosts` index.

`outputs[analytic][directories]`

Type: dict

Description: Set the default name of some important directories related to the `http_codes[active]` index.

`outputs[analytic][directories][parent]`

Type: string

Default value: `Analytic/`

Description: Set the default directory where we are going to put everything related to the HTTP analytic.

`outputs[analytic][directories][potentially_down]`

Type: string

Default value: `POTENTIALLY_INACTIVE/`

Description: Set the default directory where we are going to put all potentially inactive data.

`outputs[analytic][directories][potentially_up]`

Type: string

Default value: `POTENTIALLY_INACTIVE/`

Description: Set the default directory where we are going to put all potentially active data.

`outputs[analytic][directories][up]`

Type: string

Default value: `POTENTIALLY_INACTIVE/`

Description: Set the default directory where we are going to put all active data.

`outputs[analytic][directories][suspicious]`

Type: string

Default value: `SUSPICIOUS/`

Description: Set the default directory where we are going to put all suspicious data.

`outputs[analytic][filenames]`

Type: dict

Description: Set the default name of some important files related to the `http_codes[active]` index and the HTTP analytic subsystem.

`outputs[analytic][filenames][potentially_down]`

Type: string

Default value: down_or_potentially_down

Description: Set the default filename where we are going to put all potentially inactive data.

`outputs[analytic][filenames][potentially_up]`

Type: string

Default value: potentially_up

Description: Set the default filename where we are going to put all potentially active data.

`outputs[analytic][filenames][up]`

Type: string

Default value: active_and_merged_in_results

Description: Set the default filename where we are going to put all active data.

`outputs[analytic][filenames][suspicious]`

Type: string

Default value: suspicious_and_merged_in_results

Description: Set the default filename where we are going to put all suspicious data.

`outputs[logs]`

Type: dict

Description: Set the default name of some important files and directories related to the `logs` index.

`outputs[logs][directories]`

Type: dict

Description: Set the default name of some important directories related to the `logs` index.

`outputs[logs][directories][date_format]`

Type: string

Default value: date_format/

Description: Set the default directory where we are going to put everything related to the data when the dates are in the wrong format.

`outputs[logs][directories][no_referer]`

Type: string

Default value: no_referer/

Description: Set the default directory where we are going to put everything related to the data when no referer is found.

`outputs[logs][directories][parent]`

Type: string

Default value: no_referer/

Description: Set the default directory where we are going to put everything related to the data when no referer is found.

`outputs[logs][directories][percentage]`

Type: string

Default value: percentage/

Description: Set the default directory where we are going to put everything related to percentages.

`outputs[logs][directories][whois]`

Type: string

Default value: whois/

Description: Set the default directory where we are going to put everything related to whois data.

Note: This is the location of all files when the `debug` index is set to `True`.

`outputs[logs][filenames]`

Type: dict

Description: Set the default filenames of some important files related to the `logs` index.

`outputs[logs][filenames][auto_continue]`

Type: string

Default value: continue.json

Description: Set the default filename where we are going to put the data related to the auto continue subsystem.

Note: This file is allocated if the `auto_continue` is set to `True`.

`outputs[logs][filenames][execution_time]`

Type: `string`

Default value: `execution.log`

Description: Set the default filename where we are going to put the data related to the execution time.

Note: This file is allocated if the `show_execution_time` is set to `True`.

`outputs[logs][filenames][percentage]`

Type: `string`

Default value: `percentage.txt`

Description: Set the default filename where we are going to put the data related to the percentage.

Note: This file is allocated if the `show_percentage` is set to `True`.

`outputs[main]`

Type: `string`

Default value: `" "`

Description: Set the default location where we have to generate the `parent_directory` directory and its dependencies.

`outputs[parent_directory]`

Type: `string`

Default value: `output/`

Description: Set the directory name of the parent directory which will contain all previously nounced directories.

`outputs[splited]`

Type: `dict`

Description: Set the default name of some important files and directory related to the `split` index.

`outputs[splited][directory]`

Type: string

Default value: splited/

Description: Set the default directory name where we are going to put the split data.

16.3.59 status

Type: dict

Description: Set the needed, accepted and status name.

`status[list]`

Type: dict

Description: Set the needed and accepted status name.

Warning: All status should be in lowercase.
--

`status[list][valid]`

Type: list

Default value: ["valid", "syntax_valid", "valid_syntax"]

Description: Set the accepted VALID status.

Note: This status is only shown if the `syntax` index is activated.

`status[list][up]`

Type: list

Default value: ["up", "active"]

Description: Set the accepted ACTIVE status.

`status[list][generic]`

Type: list

Default value: ["generic"]

Description: Set the accepted generic status.

Note: This status is the one used to say the system that we have to print the complete information on the screen.

`status[list][http_active]`

Type: list

Default value: ["http_active"]

Description: Set the accepted status for the outputs[analytic][filenames][up] index.

`status[list][down]`

Type: list

Default value: ["down", "inactive", "error"]

Description: Set the accepted status INACTIVE index.

`status[list][invalid]`

Type: list

Default value: ["ouch", "invalid"]

Description: Set the accepted status INVALID index.

`status[list][potentially_down]`

Type: list

Default value: ["potentially_down", "potentially_inactive"]

Description: Set the accepted status for the outputs[analytic][filenames][potentially_down] index.

`status[list][potentially_up]`

Type: list

Default value: ["potentially_up", "potentially_active"]

Description: Set the accepted status for the outputs[analytic][filenames][potentially_up] index.

`status[list][suspicious]`

Type: list

Default value: ["strange", "hum", "suspicious"]

Description: Set the accepted status for the outputs[analytic][filenames][suspicious] index.

`status[official]`

Type: dict

Description: Set the official status name.

Note: Those status are the ones that are printed on the screen.

Warning: After any changes here please delete `dir_structure.json` and the `output/` directory.

`status[official][up]`

Type: string

Default value: ACTIVE

Description: Set the returned status for the ACTIVE case.

`status[official][down]`

Type: string

Default value: INACTIVE

Description: Set the returned status for the INACTIVE case.

`status[official][invalid]`

Type: string

Default value: INVALID

Description: Set the returned status for the INVALID case.

`status[official][valid]`

Type: string

Default value: VALID

Description: Set the returned status for the VALID case.

Note: This status is only shown if the `syntax index` is activated.

16.3.60 http_codes

Type: dict

Description: Handle the interpretation of each status codes when we do and generate our analytic data.

`http_codes[active]`

Type: boolean

Default value: True

Description: Enable / Disable the usage of the HTTP status code extraction.

`http_codes[list]`

Type: dict

Description: Categorize the http status code as mentioned in the documentation related to the HTTP Code column.

`http_codes[list][up]`

Type: list

Default value:

```
- 100
- 101
- 200
- 201
- 202
- 203
- 204
- 205
- 206
```

Description: List the HTTP status codes which are considered as ACTIVE.

`http_codes[list][potentially_down]`

Type: list

Default value:

```
- 400
- 402
- 403
- 404
- 409
- 410
- 412
- 414
- 415
- 416
```

Description: List the HTTP status code which are considered as INACTIVE or POTENTIALLY_INACTIVE.

`http_codes[list] [potentially_up]`

Type: list

Default value:

```
- 000
- 300
- 301
- 302
- 303
- 304
- 305
- 307
- 403
- 405
- 406
- 407
- 408
- 411
- 413
- 417
- 500
- 501
- 502
- 503
- 504
- 505
```

Description: List the HTTP status code which are considered as ACTIVE or POTENTIALLY_ACTIVE.

16.3.61 links

Type: dict

Description: Set the list of links which can be used/called by the system when needed.

Note: The objective of this index is to avoid hardcoded links when the configuration file is readable.

`links[api_date_format]`

Type: string

Default value: `https://pyfunceble.funilrys.com/api/date-format`

Description: Set the link to use when we share logs.

`links[api_no_referer]`

Type: string

Default value: `https://pyfunceble.funilrys.com/api/no-referer`

Description: Set the link to use when we share logs.

links[config]

Type: string

Default value: `https://raw.githubusercontent.com/funilrys/PyFunceble/master/.PyFunceble_production.yaml`

Description: Set the upstream link to the configuration file.

links[dir_structure]

Type: string

Default value: `https://raw.githubusercontent.com/funilrys/PyFunceble/master/dir_structure_production.json`

Description: Set the upstream link to the directory structure dump file.

links[iana]

Type: string

Default value: `https://raw.githubusercontent.com/funilrys/PyFunceble/master/iana-domains-db.json`

Description: Set the upstream link to the IANA zone file configuration file.

links[mariadb]

Type: string

Default value: `https://raw.githubusercontent.com/funilrys/PyFunceble/master/db_types/mariadb.sql`

Description: Set the upstream link to the MariaDB table constructor file.

links[mysql]

Type: string

Default value: `https://raw.githubusercontent.com/funilrys/PyFunceble/master/db_types/mysql.sql`

Description: Set the upstream link to the MySQL table constructor file.

links[psl]

Type: string

Default value: `https://raw.githubusercontent.com/funilrys/PyFunceble/master/public-suffix.json`

Description: Set the upstream link to the public suffix database file.

`links[repo]`

Type: `string`

Default value: `https://github.com/funilrys/PyFunceble`

Description: Set the upstream link to the repository.

`links[requirements]`

Type: `string`

Default value: `https://raw.githubusercontent.com/funilrys/PyFunceble/master/requirements.txt`

Description: Set the upstream link to the `requirements.txt` file.

17.1 From a terminal

17.1.1 `--help`

Show the help message and exit.

17.1.2 `-v` | `--version`

Show the version of PyFunceble and exit.

17.1.3 Source

`-d "something" | --domain "something"`

Test one or more domains, separated by spaces.

Note: This argument takes 1 or more values.

As example:

```
$ PyFunceble -d example.org example.net
```

Note: When this option is used, no output files are generated.

`-url "something" | --url "something"`

Test one or more full URL, separated by spaces.

Note: When we test the availability of a URL, we (only) check the HTTP status code of the given URL.

Note: This argument takes 1 or more values.

As example:

```
$ PyFunceble -u https://example.org https://example.com
```

`-f "something" | --file "something"`

Read a local or remote (RAW link) file and test all domains inside it. If remote (RAW link) file is given, PyFunceble will download it, and test the content of the given RAW link as if it was a locally stored file.

Note: The system understands the following format:

- plain list of subjects.
 - hosts file format.
-

`-uf "something" | --url-file "something"`

Read a local or remote (RAW link) file and test all (full) URLs inside it. If remote (RAW link) file is given, PyFunceble will download it, and test the content of the given RAW link as if it was a locally stored file.

Note: We consider one line as one URL to test.

Note: This argument test if a URL which is inside the given file is available. It **ONLY** tests full URLs.

As example:

```
$ PyFunceble -uf `https://raw.githubusercontent.com/funilrys/PyFunceble/master/.  
↳travis/lists/url`
```

will download the given URL and test for its content assuming that each line represents a URL to test.

Warning: A test with this argument consists of the comparison of the status code. No WHOIS record will be requested nor DNS Lookup will be done.

17.1.4 Source filtering, decoding, conversion and expansion

-ad | --adblock

Switch the decoding of the adblock format.

Default value: False

If this argument is activated the system will extract all domains or IP from the given adblock file.

--complements

Switch the value of the generation and test of the complements. A complement is for example `example.org` if `www.example.org` is given and vice-versa.

Default value: False

--filter "something"

Domain to filter (regex).

Want to test all `blogspot` from your list? This argument allows you to do that!

Note: This argument should be a regex expression.

--idna

Switch the value of the IDNA conversion.

Default value: False

This argument allows the conversion of the domains using `domain2idna`

--mining

Switch the value of the mining subsystem usage.

Default value: False

Want to find domain or URL linked to a domain in your list? This argument will exactly do that.

17.1.5 Test control

-c | --auto-continue | --continue

Switch the value of the auto continue mode.

Default value: True

This argument activates or deactivates the auto-continue subsystem. Indeed, as we can automatically continue if the script has been stopped, this switch allows us to disable or enable the usage of that specific subsystem.

`--cooldown-time`

Switch the value of the cool-down time to apply between each test.

Default value: `None`

This argument applies a number of seconds to sleep before/between each test.

`--http`

Switch the value of the usage of HTTP code.

Default value: `True`

You don't want to take the result of the HTTP code execution into consideration? This argument allows you to disable that!

Note: If activated the subsystem will bypass the HTTP status code extraction logic-representation.rst

`--local`

Switch the value of the local network testing.

Default value: `False`

Want to run a test over a local or private network? This argument will disable the limitation which does not apply to private networks.

`-ns | --no-special`

Switch the value of the usage of the SPECIAL rules.

Default value: `False`

Don't want to use/apply the SPECIAL rules - which are explained in the source column section? This argument disables them all.

`-nw | --no-whois`

Switch the value of the usage of WHOIS to test the domain's status.

Default value: `False`

Don't want to use or take into consideration the results from `whois`? This argument allows you to disable it!

`--reputation`

Switch the value of the reputation test mode.

Default value: `False`

Warning: This will disable all other forms of test, will check against AlienVault's reputation data and output its result into `output/*/{MALICIOUS, SANE}/*`.

--shadow-file | --shadow

Switch the value of the usage and generation of a shadow file before a file test starts.

A shadow file is a file which only contain the actual list of subject to test. For its generation we check each subjects as we normally do on-the-fly.

Default value: False

Note: The shadow file, will just contain the actual list of subjects to test.

--syntax

Switch the value of the syntax test mode.

Default value: False

Warning: This will disable all other forms of test, will validate the syntax of a given test subject, and output its results in plain format into `output/domains/{VALID,INVALID}/list`

-t "something" | --timeout "something"

Switch the value of the timeout in seconds.

Default value: 5

This argument will set the default timeout to apply everywhere it is possible to set a timeout.

--use-reputation-data

Switch the value of the reputation data usage.

Default value: False

Warning: This only affects when used along with the availability test.

-ua "something" | --user-agent "something"

Set the user-agent to use and set every time we interact with everything which is not the logs sharing system.

-vsc | --verify-ssl-certificate

Switch the value of the verification of the SSL/TLS certificate when testing for URL.

Default value: False

Warning: If you activate the verification of the SSL/TLS certificate, you may get **false-positive** results.

Indeed if the certificate is not registered to the CA or is simply invalid and the domain is still alive, you will always get `INACTIVE` as output.

wildcard

Switch the value of the wildcards test.

Default value: `False`

Warning: This argument is not taken into consideration if the `--syntax` argument is not given.

17.1.6 DNS (resolver) control

--dns

Set one or more specific DNS servers to use during the test. Separated by spaces.

Default value: `Follow OS DNS ==> None`

Warning: We expect a DNS server(s). If no DNS server(s) is given. You'll almost for certain get all results as `INACTIVE`

This could happen in case you use `--dns -f`

Note: You can specify a port number to use to the DNS server if needed.

As example:

```
- 127.0.1.53:5353
```

--dns-lookup-over-tcp

Make all DNS queries through TCP instead of UDP.

Default value: `False`

17.1.7 Databases

-db | --database

Switch the value of the usage of a database to store inactive domains of the currently tested list.

Default value: `True`

This argument will disable or enable the usage of a database which saves all *INACTIVE* and *INVALID* domain of the given file over time.

Note: The database is retested every x day(s), where x is the number set in `-dbr "something"`.

`--database-type`

Tell us the type of database to use. You can choose between the following: `json`, `mariadb`, `mysql`.

Default value: `json`

This argument let us use different types of database.

Note: This feature is applied to the following subsystems:

- Autocontinue physically located (JSON) at `output/continue.json`.
 - InactiveDB physically located (JSON) at `[config_dir]/inactive_db.json`.
 - Mining physically located (JSON) at `[config_dir]/mining.json`.
 - WhoisDB physically located (JSON) at `[config_dir]/whois.json`.
-

`-dbr "something" | --days-between-db-retest "something"`

Set the numbers of days between each retest of domains present into the database of *INACTIVE* and *INVALID* domains.

Default value: `1`

Note: This argument is only used if `-db` or `inactive_database : true` (under `.PyFunceble.yaml`) are activated.

`-dbc "something" | --days-between-db-clean`

Set the numbers of days since the introduction of a subject into `inactive-db.json` for it to qualifies for deletion.

Default value: `28`

Note: This argument is only used if `-db` or `inactive_database : true` (under `.PyFunceble.yaml`) are activated.

`-wdb | --whois-database`

Switch the value of the usage of a database to store whois data to avoid whois servers rate limit.

Default value: `True`

17.1.8 Output control

-a | --all

Output all available information on the screen.

Default value: False

When activated:

Domain	Status	Expiration Date	Source	HTTP Code
pyfunceble.readthedocs.io	ACTIVE	Unknown	NSLOOKUP	302

When deactivated:

Domain	Status	HTTP Code
pyfunceble.readthedocs.io	ACTIVE	302

--dots

Output dots (.) to stdout instead of giving the impression that we hang on.

Default value: False

-ex | --execution

Switch the default value of the execution time showing.

Default value: False

Want to know the execution time of your test? Well, this argument will let you know!

--hierarchical

Switch the value of the hierarchical sorting of the tested file.

Default value: True

This argument will output the result listed in a hierarchical order.

-h | --host

Switch the value of the generation of hosts file.

Default value: True

This argument will let the system know if it has to generate the hosts file version of each status.

-ip "something"

Change the IP to print in the hosts files with the given one.

Default value: 0.0.0.0

--json

Switch the value of the generation of the JSON formatted list of domains.

Default value: False

--less**When activated:**

Domain	Status	HTTP Code
pyfuncible.readthedocs.io	ACTIVE	302

When deactivated:

Domain	Status	Expiration Date	Source	HTTP Code
pyfuncible.readthedocs.io	ACTIVE	Unknown	NSLOOKUP	302

-nf | --no-files

Switch the value of the production of output files.

Default value: False

Want to disable the production of the outputted files? This argument is for you!

-nl | --no-logs

Switch the value of the production of logs files in the case we encounter some errors.

Default value: False

Don't want any logs to go out of PyFuncible? This argument disables every log subsystem.

-nu | --no-unified

Switch the value of the production unified logs under the output directory.

Default value: True

This argument disables the generation of *result.txt*.

--percentage

Switch the value of the percentage output mode.

Default value: True

This argument will disable or enable the generation of the percentage of each status.

`--plain`

Switch the value of the generation of the plain list of domains.

Default value: `False`

Want to get a list with all domains for each status? The activation of this argument does the work while testing!

`-q | --quiet`

Run the script in quiet mode.

Default value: `False`

You prefer to run a program silently? This argument is for you!

`--share-logs`

Switch the value of the sharing of logs.

Default value: `False`

Want to make PyFuncible a better tool? Share your logs with our API which collect all logs!

`-s | --simple`

Switch the value of the simple output mode.

Default value: `False`

Want as less as possible data on screen? This argument returns as less as possible on screen!

`--split`

Switch the value of the split of the generated output

Default value: `True`

Want to get the logs (copy of what you see on screen) on different files? This argument is suited to you!

17.1.9 Multiprocessing

`-m | --multiprocess`

Switch the value of the usage of multiple processes.

Default value: `False`

Want to speed up the test time? This argument will allow the usage of multiple processes for testing.

--multiprocess-merging-mode

Sets the multiprocess merging mode. You can choose between the following *livelends*.

Default value: `end`

Note: With the `end` value, the merging of cross-process data is made at the very end of the current instance.

Note: With the `live` value, the merging of cross-process data is made after the processing of the maximal number of processes.

Which means that if you allow 5 processes, we will run 5 tests, merge, run 5 tests, merge and so on until the end.

-p | --processes

Set the number of simultaneous processes to use while using multiple processes.

Default value: `25`

Warning: DO not try to exceed your number of CPU if you want to keep your machine somehow alive and healthy!!

Note: If omitted, the number of available CPU cores will be used instead.

17.1.10 CI / CD

--autosave-minutes

Update the minimum of minutes before we start committing to upstream under the CI mode.

Default value: `15`

--ci

Switch the value of the CI mode.

Default value: `False`

Note: If you combine this argument with the `--quiet` argument, the test will output a dotted line, where each dot (.) represent one test result or input which was skipped because it was previously tested.

Want to use PyFuncible under a supporter CI infrastructure/network? This argument is suited for your needs!

`--ci-branch`

Switch the branch name where we are going to push the temporary results.

Default value: `master`

Note: Currently the branch need to exist, but there are being worked on a path to have PyFunceble to create the sub-branch and finally merge it into the `--ci-distribution-branch`

`--ci-distribution-branch`

Switch the branch name where we are going to push the final results.

Default value: `master`

Note: The difference between this and `--ci-branch` is the fact that this branch will get the (final) result only when the test is finished under the given `--ci-branch`.

As an example, this allows us to have 2 branches:

- `processing` (CI branch), for the tests with PyFunceble.
 - `master` (CI distribution branch), for the distribution of the results of PyFunceble.
-

`--cmd "something"`

Pass a command before each commit (except the final one).

Default value: `' '`

Note: In this example, `something` should be a script or a program which have to be executed when we reached the end of the given file.

Note: This argument is only used if `--ci` or `ci: true` (under `.PyFunceble.yaml`) are activated.

`--cmd-before-end "something"`

Pass a command before the results (final) commit under the CI mode.

Default value: `' '`

Note: In this example, `something` should be a script or a program which have to be executed when we reached the end of the given file.

Note: This argument is only used if `--ci` or `ci: true` (under `.PyFunceble.yaml`) are activated.

`--commit-autosave-message "something"`

Replace the default autosave commit message.

Default value: `PyFunceble - AutoSave`

This argument allows us to set a custom commit message which is going to be used as a commit message when saving.

Note: This argument is only used if `--ci` or `ci: true` (under `.PyFunceble.yaml`) are used.

Note: This argument is only used if we have to split the work into multiple processes because a list is too long or the timeout is reached.

Warning: Please avoid the usage of `[ci skip]` here.

`--commit-results-message "something"`

Replace the default results (final) commit message.

Default value: `PyFunceble - Results`

Note: This argument is only used if `--ci` or `ci: true` (under `.PyFunceble.yaml`) are used.

Note: This argument is only used if we reached the end of the list we are or have to test.

17.1.11 Unique actions

`--clean`

Clean all files under the output directory.

As it is sometimes needed to clean our `output/` directory, this argument does the job automatically.

Warning: This argument delete everything which are `.keep` or `.gitignore`

`--clean-all`

Clean all files under the output directory along with all file generated by PyFunceble.

Warning: This deletes almost everything we generated without any warning.

Note: We don't delete the whois database file/table because they are (almost) static data which are shared across launches in your environment.

Warning: If you plan to clean manually do not delete the whois database file or table as it will make your test finish under a much longer time as usual for you.

Warning: If you don't combine this argument alongside with the `--database-type` argument or its configurable equivalent, this argument will only clean the JSON formatted databases.

`--directory-structure`

Generate the directory and files that are needed and which does not exist in the current directory.

Want to start without anything? This argument generates the output directory automatically for you!

Note: In case of a file or directory not found issue, it's recommended to remove the `dir_structure.json` along with the `output/` directory before using this argument.

17.1.12 Global overview

```
usage: pyfunceble [-d DOMAIN [DOMAIN ...]] [-u URL [URL ...]] [-f FILE]
                  [-uf URL_FILE] [-ad] [--complements] [--filter FILTER]
                  [--idna] [--mining] [-c] [--cooldown-time COOLDOWN_TIME]
                  [--http] [--local] [-ns] [-nw] [--syntax] [-t TIMEOUT]
                  [--reputation] [--use-reputation-data] [-ua USER_AGENT]
                  [-vsc] [--wildcard] [--dns DNS [DNS ...]]
                  [--dns-lookup-over-tcp] [-db]
                  [--database-type DATABASE_TYPE]
                  [-dbr DAYS_BETWEEN_DB_RETEST] [-dbc DAYS_BETWEEN_DB_CLEAN]
                  [-wdb] [-a] [-ex] [--hierarchical] [-h] [-ip IP] [--json]
                  [--less] [-nf] [-nl] [-nu] [--percentage] [--plain] [--dots]
                  [-q] [--share-logs] [-s] [--split] [-m]
                  [--multiprocess-merging-mode MULTIPROCESS_MERGING_MODE]
                  [-p PROCESSES] [--autosave-minutes AUTOSAVE_MINUTES] [--ci]
                  [--ci-branch CI_BRANCH]
                  [--ci-distribution-branch CI_DISTRIBUTION_BRANCH]
                  [--cmd CMD] [--cmd-before-end CMD_BEFORE_END]
                  [--commit-autosave-message COMMIT_AUTOSAVE_MESSAGE]
                  [--commit-results-message COMMIT_RESULTS_MESSAGE] [--clean]
                  [--clean-all] [--directory-structure] [--help] [-v]
```

PyFunceble - The tool to check the availability or syntax of domain, IP or URL.

optional arguments:

<code>--help</code>	Show this help message and exit.
<code>-v, --version</code>	Show the version of PyFunceble and exit.

(continues on next page)

(continued from previous page)

```

Source:
  -d DOMAIN [DOMAIN ...], --domain DOMAIN [DOMAIN ...]
                                Test one or more domains, separated by spaces.

                                When this option is used, no output files are generated.
  -u URL [URL ...], --url URL [URL ...]
                                Test one or more full URL, separated by spaces.
  -f FILE, --file FILE Read a local or remote (RAW link) file and test all domains
↳inside it.
                                If remote (RAW link) file is given, PyFunceble will
↳download it,
                                and test the content of the given RAW link as if it was a
↳locally stored file.
  -uf URL_FILE, --url-file URL_FILE
                                Read a local or remote (RAW link) file and test all
↳(full) URLs inside it.
                                If remote (RAW link) file is given, PyFunceble will
↳download it,
                                and test the content of the given RAW link as if it was a
↳locally stored file.

                                This argument test if an URL is available. It ONLY test
↳full URLs.

Source filtering, decoding, conversion and expansion:
  -ad, --adblock Switch the decoding of the adblock format.
                                Configured value: False
  --complements Switch the value of the generation and test of the
↳complements.
                                A complement is for example `example.org` if `www.example.
↳org` is given and vice-versa.
                                Configured value: False
  --filter FILTER Domain to filter (regex).
  --idna Switch the value of the IDNA conversion.
                                Configured value: False
  --mining Switch the value of the mining subsystem usage.
                                Configured value: False

Test control:
  -c, --auto-continue, --continue
                                Switch the value of the auto continue mode.
                                Configured value: True
  --cooldown-time COOLDOWN_TIME
                                Switch the value of the cooldown time to apply between
↳each test.
                                Configured value: None
  --http Switch the value of the usage of HTTP code.
                                Configured value: True
  --local Switch the value of the local network testing.
                                Configured value: True
  -ns, --no-special Switch the value of the usage of the SPECIAL rules.
                                Configured value: False
  -nw, --no-whois Switch the value of the usage of WHOIS to test the domain's
↳status.
                                Configured value: False
  --syntax Switch the value of the syntax test mode.
                                Configured value: False

```

(continues on next page)

(continued from previous page)

```

-t TIMEOUT, --timeout TIMEOUT
    Switch the value of the timeout in seconds.
    Configured value: 5
--reputation
    Switch the value of the reputation test mode.
    Configured value: False
--use-reputation-data
    Switch the value of the reputation data usage.
    Configured value: False
-ua USER_AGENT, --user-agent USER_AGENT
    Set the user-agent to use and set every time we interact
↳with everything which
    is not the logs sharing system.
-vsc, --verify-ssl-certificate
    Switch the value of the verification of the SSL/TLS
↳certificate when testing for URL.
    Configured value: False
--wildcard
    Switch the value of the wildcards test.

    When used, wildcards will be properly tested.

    Warning: This is not taken in consideration if the '--
↳syntax' argument is not given.
    Configured value: False

DNS (resolver) control:
--dns DNS [DNS ...]
    Set one or more DNS server(s) to use during testing.
↳Separated by spaces.

    To specify a port number for the DNS server you append it
↳as :port [ip:port].

    If no port is specified, the default DNS port (53) is
↳used.
    Configured value: OS (declared) DNS server
--dns-lookup-over-tcp
    Make all DNS queries with TCP.
    Configured value: False

Databases:
-db, --database
    Switch the value of the usage of a database to store
↳inactive domains of the currently tested list.
    Configured value: True
--database-type DATABASE_TYPE
    Tell us the type of database to use.
    You can choose between the following: `json | mariadb |
↳mysql`
    Configured value: 'json'
-dbr DAYS_BETWEEN_DB_RETEST, --days-between-db-retest DAYS_BETWEEN_DB_RETEST
    Set the numbers of days between each retest of domains
↳present into inactive-db.json.
    Configured value: 1
-dbc DAYS_BETWEEN_DB_CLEAN, --days-between-db-clean DAYS_BETWEEN_DB_CLEAN
    Set the numbers of days since the introduction of a
↳subject into inactive-db.json for it to qualifies for deletion.
    Configured value: 28
-wdb, --whois-database
    Switch the value of the usage of a database to store
↳whois data to avoid whois servers rate limit.

```

(continues on next page)

(continued from previous page)

```

Configured value: True

Output control:
  -a, --all          Output all available information on the screen.
                     Configured value: True
  -ex, --execution   Switch the default value of the execution time showing.
                     Configured value: False
  --hierarchical     Switch the value of the hierarchical sorting of the tested_
↳file.
                     Configured value: False
  -h, --host         Switch the value of the generation of hosts file.
                     Configured value: True
  -ip IP             Change the IP to print in the hosts files with the given_
↳one.
                     Configured value: '0.0.0.0'
  --json             Switch the value of the generation of the JSON formatted_
↳list of domains.
                     Configured value: False
  --less             Output less informations on screen.
                     Configured value: False
  -nf, --no-files    Switch the value of the production of output files.
                     Configured value: False
  -nl, --no-logs     Switch the value of the production of logs files in the_
↳case we encounter some errors.
                     Configured value: False
  -nu, --no-unified  Switch the value of the production unified logs under the_
↳output directory.
                     Configured value: False
  --percentage       Switch the value of the percentage output mode.
                     Configured value: True
  --plain            Switch the value of the generation of the plain list of_
↳domains.
                     Configured value: False
  --dots             Prints dots to stdout instead of giving the impression that_
↳we hang on.
                     Configured value: False
  -q, --quiet        Run the script in quiet mode.
                     Configured value: False
  --share-logs       Switch the value of the sharing of logs.
                     Configured value: False
  -s, --simple        Switch the value of the simple output mode.
                     Configured value: False
  --split            Switch the value of the split of the generated output files.
                     Configured value: True

Multiprocessing:
  -m, --multiprocess Switch the value of the usage of multiple processes.
                     Configured value: False
  --multiprocess-merging-mode MULTIPROCESS_MERGING_MODE
                     Sets the multiprocess merging mode.
                     You can choose between the following: `live|ends`.
                     Configured value: 'end'
  -p PROCESSES, --processes PROCESSES
                     Set the number of simultaneous processes to use while_
↳using multiple processes.
                     If omitted, the number of available CPU cores will be used_
↳instead.

```

(continues on next page)

(continued from previous page)

```

Configured value: 25

CI / CD:
  --autosave-minutes AUTOSAVE_MINUTES
    Update the minimum of minutes before we start committing.
    ↳to upstream under the CI mode.
    Configured value: 15
  --ci
    Switch the value of the CI mode.
    Configured value: False
  --ci-branch CI_BRANCH
    Switch the branch name where we are going to push the
    ↳temporary results.
    Configured value: 'master'
  --ci-distribution-branch CI_DISTRIBUTION_BRANCH
    Switch the branch name where we are going to push the
    ↳final results.
    Configured value: 'master'
  --cmd CMD
    Pass a command to run before each commit (except the final
    ↳one) under the CI mode.
    Configured value: ''
  --cmd-before-end CMD_BEFORE_END
    Pass a command to run before the results (final) commit
    ↳under the CI mode.
    Configured value: ''
  --commit-autosave-message COMMIT_AUTOSAVE_MESSAGE
    Replace the default autosave commit message.
    Configured value: None
  --commit-results-message COMMIT_RESULTS_MESSAGE
    Replace the default results (final) commit message.
    Configured value: None

Unique actions:
  --clean
    Clean all files under the output directory.
  --clean-all
    Clean all files under the output directory along with all
    ↳file generated by PyFuncible.
  --directory-structure
    Generate the directory and files that are needed and
    ↳which does not exist in the current directory.

For an in-depth usage, explanation and examples of the arguments, you should read the
↳documentation at https://pyfuncible.readthedocs.io/en/master/

Crafted with by Nissar Chababy (@funilrys) with the help of https://pyfuncible.github.io/contributors.html && https://pyfuncible.github.io/special-thanks.html

```

17.2 From a GitLab CI/CD environment

As we offer an argument named `--ci` which will autosave in a GitLab CI/CI environment, this document try to describe how it works!

17.2.1 Configuration

Personal Access Token

A personal access token is needed in order for PyFunceble to automatically push the results.

You should get a [personal GitLab access token](#) with the `read_repository` and `write_repository` scopes.

Once created and copied in a safe place, create a **new masked variable** named `GL_TOKEN` inside the CI/CD settings of your project. The value of the variable should be the newly generated personal access token.

`.gitlab-ci.yml`

Note: This part only present a commented `.gitlab-ci.yml`. This is just an example do not take the following as necessarily true.

You're invited to submit changes if something stated in this document is wrong.

```
# Python needed, so we use the python image.
image: python:latest

variables:
  # This is the Git name we have to set. (git config user.name)
  GIT_EMAIL: "dead-hosts@funilrys.com"
  # This is the Git Email we have to set. (git config user.email)
  GIT_NAME: "GitLab CI/CD"

before_script:
  # We install the development version of PyFunceble.
  # If you prefer the stable version replace `pyfunceble-dev`
  # with `pyfunceble`.
  - pip3 install PyFunceble-dev

run:
  script:
    # Let's say we want our results and our PyFunceble
    # infrastructure to be saved in a directory called `PyFunceble-tests`

    # We move inside it.
    - cd PyFunceble-tests
    # We test the file `my_awesome_list` which is located inside the current_
    ↪directory.
    # Note: we precise the `--ci` argument here,
    # but you work without it if you set `ci: true` inside your `.PyFunceble.
    ↪yaml`
    - PyFunceble --ci -f my_awesome_list --plain
```

17.3 From a Travis CI container

As we offer an argument named `--ci` to activate the usage of PyFunceble in a Travis CI instance, we document here what you need to know!

17.3.1 Configuration

Note: This part only present a commented `.travis.yml` so that you can understand where to start.

If you need more practical examples, feel free to report to one of [Dead-Hosts](#) repositories which use PyFunceble with Travis CI.

```
env:
  global:
    # The following is your encrypted GitHub API key.
    # Indeed as we are going to push to the repository, this is needed.
    #- GH_TOKEN: # This can be set in the travis-ci https://travis-ci.com/repo/
    ↪ settings as 'Environment Variables'
    # or as below: secure: encrypted code
    - secure: QQdKFquFFoJFT9XJ1XZp4EMoDTVoXFgqZq8XU+sCVf+pJQR6d/
    ↪ oKBp8rnSTCnZizWOQXUjGXUUxUpSG/
    ↪ dYGYBLjo3rH3rsn9ciZHVfubxbwK860w4sqibl4DvhCv2rdsFtvzXnhm4P9OL3i+krKdewh9fxpNyUU58qOgfnS7mK9FcFhb8z
    ↪ dc0y36q6vjfgJR+QchetOtHgNbKYbLB8c26Di90OZCFJsxMNcl1Wct4qFPXkFGvjXrISW6pbdPL5Plto0Ig3iLiulhYOPVArysl
    ↪ 67d+F9ocQdIoFpDDTDgIjHerQnD2wRglaKPzLDb4jJTpqgr5ssPrqUAKl3st7gyaAZzCEADPDnIBDjOJS+mFWbX9DKgc=
    # This is the Git name we have to set. (git config user.name)
    - GIT_NAME: Travis CI
    # This is the Git Email we have to set. (git config user.email)
    - GIT_EMAIL: dead-hosts@funilrys.com

# This is the language we use.
language: python

# This is the python version we are going to use for the tests.
# Note: you can add any 3.x version to the list.
python:
  - "3.6"

# The following will tell Travis CI to ends as fast as possible.
matrix:
  fast_finish: true

# Here we are setting what Travis CI have to cache.
cache:
  # We are caching pip3 as we use it to install PyFunceble
  - pip3

install:
  # We install the development version of PyFunceble. If you prefer the stable_
  ↪ version replace
  # `pyfunceble-dev` with `pyfunceble`.
  - pip3 install pyfunceble-dev

# Our tests start here.
script:
  # Let's say we want our results and our PyFunceble infrastructure to be saved in_
  ↪ a directory
  # called `PyFunceble-tests`

  # We move inside it.
  - cd PyFunceble-tests
  # We test the file `my_awesome_list` which is located inside the current_
  ↪ directory.
  # Note: we precise the `--ci` argument here,
```

(continues on next page)

(continued from previous page)

```

# but you work without it if you set `travis: true` inside your `.PyFunceble.
↪yaml`
- PyFunceble --ci -f my_awesome_list --plain

# The following initiate email notification logic.
notifications:
  # As we want to get a mail on failure and on status change, we set the following.
  on_success:  change
  on_failure:  always

```

17.3.2 Getting a GitHub token

For the secure index of the `.travis.yml` file, you have to generate a new [GitHub token](#).

After you got your token, please write it or save it in a safe place as you're going to need it every time you're going to interact with Travis CI.

Note: The scope to set is `public_repo` but you can also set others depending on your needs.

17.3.3 Encrypting the token for future usage under the Travis CIs' containers

To encrypt the token simply replace and execute the following according to your personal case.

```

$ travis encrypt 'GH_TOKEN=theGeneratedToken' -r 'The content of TRAVIS_REPO_SLUG' --
↪add

```

Warning: Please do not execute the following explicitly without replacing `theGeneratedToken` with your previously generated GitHub token and `The content of TRAVIS_REPO_SLUG` with your repository slug.

Note: The usage of `--add` ensure that the `travis` program automatically add the secure index to the `.travis.yml` file.

17.4 Using the PyFunceble (Python) API

If you are working with a python script, module or even class, you can integrate **PyFunceble** to your main logic by importing it and using its API (cf: [API Documentation](#)).

This section will present some example of the way you can interact with PyFunceble from anything written in Python.

17.4.1 Get the availability of domains or IP

Note: This example can be found in our [examples repository](#).

Todo: Add IPs in the loop.

```
"""
This is an example which respond to the following problematic(s):

    * How can I get the availability of a domain or IP with PyFunceble ?
"""

# We want some coloration so we import the tool do to that :)
from PyFunceble import initiate_colorama, Fore, Style
# We import the tool to print the colored CLI logo.
from PyFunceble.cli_core import CLICore
# We import the configuration loader.
from PyFunceble import load_config
# We import the test method of the PyFunceble API.
from PyFunceble import test as PyFunceble

# We initiate the list of domains we are going to test.
DOMAINS = [
    "google.com",
    "tweeetttter.com",
    "github.com",
    "examplessss.ooooorgg",
    "twitter.com",
    "forest-jump"
]

# We initiate colorama.
initiate_colorama(True)

# We load our configuration.
#
# Note: We need this to print the logo but if you
# doesn't need the logo, you can ignore this.
load_config(generate_directory_structure=False)

# We print the PyFunceble logo.
CLICore.colorify_logo(home=True)

def print_result(subject, status):
    """
    Given the subject and its status, we print it to STDOUT.

    :param str subject: The subject we are going to print.
    :param str status: The status of the domain.
    """

    if status == "ACTIVE":
        print(f"{Fore.GREEN + Style.BRIGHT}{domain} is {status}")
    elif status == "INACTIVE":
        print(f"{Fore.RED + Style.BRIGHT}{domain} is {status}")
    else:
        print(f"{Fore.CYAN + Style.BRIGHT}{domain} is {status}")

for domain in DOMAINS:
    # We loop through the list of domain.
```

(continues on next page)

(continued from previous page)

```
# And we print the domain and status with the right coloration!
print_result(domain, PyFunceble(domain))
```

17.4.2 Get the availability of URL

Note: This example can be found in our [examples repository](#).

```
"""
This is an example which respond to the following problematic(s):

    * How can I get the availability of an URL with PyFunceble ?
"""

# We want some coloration so we import the tool do to that :)
from PyFunceble import initiate_colorama, Fore, Style
# We import the tool to print the colored CLI logo.
from PyFunceble.cli_core import CLICore
# We import the configuration loader.
from PyFunceble import load_config
# We import the test method of the PyFunceble API.
from PyFunceble import url_test as PyFunceble

# We initiate the list of URLs we are going to test.
URLS = [
    "https://google.com",
    "http://tweeetttter.com",
    "ftp://github.com",
    "http://examplessss.ooooorgg",
    "http://twitter.com",
]

# We initiate colorama.
initiate_colorama(True)

# We load our configuration.
#
# Note: We need this to print the logo but if you
# doesn't need the logo, you can ignore this.
load_config(generate_directory_structure=False)

# We print the PyFunceble logo.
CLICore.colorify_logo(home=True)

def print_result(subject, status):
    """
    Given the subject and its status, we print it to STDOUT.

    :param str subject: The subject we are going to print.
    :param str status: The status of the domain.
    """

    if status == "ACTIVE":
```

(continues on next page)

(continued from previous page)

```
    print(f"{Fore.GREEN + Style.BRIGHT}{domain} is {status}")
    elif status == "INACTIVE":
        print(f"{Fore.RED + Style.BRIGHT}{domain} is {status}")
    else:
        print(f"{Fore.CYAN + Style.BRIGHT}{domain} is {status}")

for url in URLs:
    # We loop through the list of domain.

    # And we print the domain and status with the right coloration!
    print_result(url, PyFunceble(url))
```

17.4.3 Complete dataset while getting the availability of domains, IPs or URL

While using our API, you can request to see/get everything with the help of the `complete=True` argument.

You'll then get the following dict as output.

```
{
    "_status": None, # If some extra rules are applied, this index will keep the_
    ↪ status before the extra rules was applied.
    "_status_source": None, # If some extra rules are applied, this index will keep_
    ↪ the source before the extra rules was applied.
    "domain_syntax_validation": None, # The domain syntax validation status.
    "expiration_date": None, # The expiration date of the tested subject (if found).
    "http_status_code": None, # The status code of the tested subejct.
    "ip4_syntax_validation": None, # The IPv4 syntax validation status.
    "dns_lookup": [], # The DNS Lookup output.
    "status_source": None, # The (final) source which gave us the status.
    "status": None, # The (final) status returned by PyFunceble.
    "tested": None, # The tested subject.
    "url_syntax_validation": None, # The url syntax validation status.
    "whois_record": None, # The whois record (if found).
    "whois_server": None, # The whois server we use to get the whois record (if_
    ↪ found).
}
```

17.4.4 Set custom configuration index while getting the availability of domains, IPs or URL

While using PyFunceble, you might want to set or overwrite a default behaviour.

You can do that in 2 ways. Globally or locally.

17.4.5 Globally

To set globally simply initiate the configuration loader and parse your custom configuration along with the initialization.

As example, you can do it like follow:


```
# We import the configuration loader.
from PyFunceble import load_config

# We set our list of indexes to overwrite.
OUR_PYFUNCEBLE_CONFIG = {"share_logs":False, "no_files": True}

# We load our configuration and parse our custom indexes.
load_config(generate_directory_structure=False, custom=OUR_PYFUNCEBLE_CONFIG)

## We can then play with PyFunceble and/or other business logic ...
```

Locally

To set globally simply parse your configuration along with the test method.

As example, you can do it like follow:

```
# We import the test method.
from PyFunceble import test as AvailabilityTest

# We set our list of indexes to overwrite.
OUR_PYFUNCEBLE_CONFIG = {"share_logs":False, "no_files": True}

# We get the status and parse our configuration.
status = AvailabilityTest("hello.world", config=OUR_PYFUNCEBLE_CONFIG)

## We can then manipulate the status and/or other business logic ...
```

17.4.6 Check the syntax of domains

Note: This example can be found in our [examples repository](#).

```
"""
This is an example which respond to the following problematic(s):

    * How can I check the syntax of a domain with PyFunceble ?
"""

# We want some coloration so we import the tool do to that :)
from PyFunceble import initiate_colorama, Fore, Style
# We import the tool to print the colored CLI logo.
from PyFunceble.cli_core import CLICore
# We import the configuration loader.
from PyFunceble import load_config
# We import the test method of the PyFunceble API.
from PyFunceble import is_domain as PyFunceble

# We initiate the list of domains we are going to test.
DOMAINS = [
    "google.com",
    "tweetttiter.com",
    "github.com",
```

(continues on next page)

(continued from previous page)

```

    "examplessss.ooooorgg",
    "twitter.com",
    "forest-jump",
]

# We initiate colorama.
initiate_colorama(True)

# We load our configuration.
#
# Note: We need this to print the logo but if you
# doesn't need the logo, you can ignore this.
load_config(generate_directory_structure=False)

# We print the PyFunceble logo.
CLICore.colorify_logo(home=True)

def print_syntax_result(subject, status):
    """
    Given the subject and its validation, we print it to STDOUT.

    :param str subject: The subject we are going to print.
    :param bool status: The validation state.
    """

    if status is True:
        print(f"{Fore.GREEN + Style.BRIGHT}{subject} is VALID")
    else:
        print(f"{Fore.CYAN + Style.BRIGHT}{subject} is INVALID")

for domain in DOMAINS:
    # We loop through the list of domain.

    # And we print the domain and status with the right coloration!
    print_syntax_result(domain, PyFunceble(domain))

```

17.4.7 Check the syntax of IPv4s

Note: This example can be found in our [examples repository](#).

```

"""
This is an example which respond to the following problematic(s):

    * How can I check the syntax of an IPv4/IPv6 with PyFunceble ?
"""

# We want some coloration so we import the tool do to that :)
from PyFunceble import initiate_colorama, Fore, Style
# We import the tool to print the colored CLI logo.
from PyFunceble.cli_core import CLICore
# We import the configuration loader.
from PyFunceble import load_config

```

(continues on next page)

(continued from previous page)

```

# We import the test method of the PyFunceble API.
from PyFunceble import is_ip as PyFunceble

# We initiate the list of IPs we are going to test.
IPS = ["216.58.207.46", "257.58.207.46"]

# We initiate colorama.
initiate_colorama(True)

# We load our configuration.
#
# Note: We need this to print the logo but if you
# doesn't need the logo, you can ignore this.
load_config(generate_directory_structure=False)

# We print the PyFunceble logo.
CLICore.colorify_logo(home=True)

def print_syntax_result(subject, status):
    """
    Given the subject and its validation, we print it to STDOUT.

    :param str subject: The subject we are going to print.
    :param bool status: The validation state.
    """

    if status is True:
        print(f"{Fore.GREEN + Style.BRIGHT}{subject} is VALID")
    else:
        print(f"{Fore.CYAN + Style.BRIGHT}{subject} is INVALID")

for ip in IPS:
    # We loop through the list of IP.

    # And we print the IP and status with the right coloration!
    print_syntax_result(ip, PyFunceble(ip))

```

17.4.8 Check the syntax of URLs

Note: This example can be found in our [examples repository](#).

```

"""
This is an example which respond to the following problematic(s):

    * How can I check the syntax of an URL with PyFunceble ?
"""

# We want some coloration so we import the tool do to that :)
from PyFunceble import initiate_colorama, Fore, Style
# We import the tool to print the colored CLI logo.
from PyFunceble.cli_core import CLICore
# We import the configuration loader.

```

(continues on next page)

(continued from previous page)

```
from PyFunceble import load_config
# We import the test method of the PyFunceble API.
from PyFunceble import is_url as PyFunceble

# We initiate the list of URLs we are going to test.
URLS = [
    "https://google.com",
    "http://tweeetttter.com",
    "http://github.com",
    "http://examplessss.ooooorgg",
    "https://twitter.com",
    "http:forest-jump",
]

# We initiate colorama.
initiate_colorama(True)

# We load our configuration.
#
# Note: We need this to print the logo but if you
# doesn't need the logo, you can ignore this.
load_config(generate_directory_structure=False)

# We print the PyFunceble logo.
CLICore.colorify_logo(home=True)

def print_syntax_result(subject, status):
    """
    Given the subject and its validation, we print it to STDOUT.

    :param str subject: The subject we are going to print.
    :param bool status: The validation state.
    """

    if status is True:
        print(f"{Fore.GREEN + Style.BRIGHT}{subject} is VALID")
    else:
        print(f"{Fore.CYAN + Style.BRIGHT}{subject} is INVALID")

for url in URLS:
    # We loop through the list of URL.

    # And we print the URL and status with the right coloration!
    print_syntax_result(url, PyFunceble(url))
```

18.1 Subject

Give us the tested element.

Note: The API equivalent is `tested`.

18.2 Status

Give us the test result/status of the tested element.

Note: The API equivalent is `status`.

18.2.1 ACTIVE

This status is returned when **one of the following cases** is met:

- We can extract the expiration date from `PyFunceble.lookup.whois.WhoisLookup.request()`.

Note: We don't check if the extracted date is in the past. Indeed, if we start to do the registry work, it becomes a never ending work for us.

- `PyFunceble.lookup.dns.DNSLookup.request()` does not return `{}`.

Note: We don't read nor interpret the returned data.

As the request `PyFunceble.lookup.dns.DNSLookup.request()` method already handle everything, we don't need to read the given data.

We just check that it is not an empty dictionary.

- `PyFunceble.lookup.dns.DNSLookup.request()` returns {}, `PyFunceble.lookup.whois.WhoisLookup.request()` provides nothing exploitable, but `PyFunceble.lookup.http_code.HTTPCode.get()` returned something which is not the default value (XXX).

18.2.2 INACTIVE

This status is returned when **all the following cases** are met:

- We could not extract the expiration date from `PyFunceble.lookup.whois.WhoisLookup.request()`.
- `PyFunceble.lookup.dns.DNSLookup.request()` returns nothing.
- `PyFunceble.lookup.http_code.HTTPCode.get()` is not in the list of ACTIVE status codes.

18.2.3 INVALID

This status is returned when **all the following cases** are met:

- The Domain/IP does not match/pass our syntax checker.
- The domain extension is unregistered in the [IANA Root Zone Database](#), our internal list nor in the [Public Suffix List](#).

Note: Understand by this that the extension is not present:

- in the `iana-domains-db.json` file
 - in the `public-suffix.json` file
 - in the `PyFunceble.check.Check.SPECIAL_USE_DOMAIN_NAMES_EXTENSIONS` attribute.
-

18.2.4 VALID

This status is returned when we are checking for syntax. It is the equivalent of ACTIVE but for syntax checking.

18.2.5 MALICIOUS

This status is returned when we are checking for the reputation.

18.2.6 SANE

This status is returned when we are checking for the reputation.

18.3 Source

Give us the source of the status.

Note: The API equivalent is `status_source`.

18.3.1 HTTP Code

This source is returned when **all the following cases** are met:

- We can't extract the expiration date from `PyFunceble.lookup.whois.WhoisLookup.request()`.
- The `INACTIVE` status is the one returned by other methods.
- `PyFunceble.lookup.http_code.HTTPCode.get()` outputs is different from the default one (`XXX`) and the other methods gives the `INACTIVE` status.

18.3.2 SYNTAX

This source is always returned when the domain has the status `INVALID` or in the case that we are only checking for syntax instead of availability. The usage of this source comes from the comparison of the element against our domain, IP or URL syntax validation system.

18.3.3 DNSLOOKUP

This source is always returned when the taken decision of the status of the domain/IP comes from `PyFunceble.lookup.dns.DNSLookup.request()` outputs.

18.3.4 SPECIAL

As `PyFunceble` grows, I thought that a bit of filtering for special cases would be great. So I introduced the `SPECIAL` source.

Note: Please consider all 3 digits number that are listed in this section as the HTTP status code caught by `PyFunceble.lookup.http_code.HTTPCode.get()`.

Warning: Do not want those rules ? You can use following to disable them.

- `-ns|--no-special` arguments from the CLI.
- `no_special: True` into your local configuration file.

.blogspot.

- All 404 are returned as `INACTIVE`
- All 301 which are blocked by Google or does not exist are returned as `INACTIVE`

- All 302 which are blocked by Google are returned as INACTIVE

***.canalblog.com**

- All 404 are returned as INACTIVE

***.github.io**

- All 404 are returned as INACTIVE

***.liveadvert.com**

- All 404 are returned as INACTIVE

***.skyrock.com**

- All 404 are returned as INACTIVE

***.tumblr.com**

- All 404 are returned as INACTIVE

***.wordpress.com**

- All 301 which match `doesn't exist` are returned as INACTIVE

IP with range

- All IPv4 with a range (for example `0.0.0.0/24`) are returned as ACTIVE
- All IPv6 with a range (for example `2001:db8::/43`) are returned as ACTIVE

Reputation

Note: If the `--use-reputation-data` argument is activated or the `use_reputation_data` index of your configuration file is active, the following apply.

- All IPv4 and IPv6 which are present into the AlienVault public reputation data are returned as ACTIVE

18.4 __status (API)

This index has the same proprieties as `status` but before any extra rules were applied.

18.5 `_status_source` (API)

This index has the same proprieties as `status_source` but before any extra rules were applied.

18.6 `dns_lookup` (API)

Provides the output of the DNS lookup.

18.6.1 Format 1: Normal DNS Lookup

The first and most common output will be like follow:

```
{
  "A": ["string"],
  "AAAA": ["string"],
  "CNAME": ["string"],
  "NS": ["string"],
  "PTR": ["string"]
}
```

where each list item represents the response from the DNS server.

18.6.2 Format 2: `get_hosts_by_addr` (local subjects?)

For the system to not miss something, it will try to execute the `socket.gethostbyaddr()` (for IP) and `socket.gethostbyname()` (for the rest). In both cases, the result will be like follow:

```
{
  "hostname": "string",
  "aliases": ["string"],
  "ips": ["string"]
}
```

18.7 HTTP Code

Note: The API equivalent is `http_status_code`.

Note: This section reflects exactly what the code does. So any changes made here should also be reported to the code or at least the configuration file.

Note: A XXX in this column means that it was impossible to catch the HTTP status code from the web server.

Note: The Status Codes we give to PyFunceble to test with can be fully customized in your own `.PyFunceble.yaml`.

18.7.1 As active

Note: While testing for domain and IP, a subject which has an HTTP code listed below, will be saved/logged into the `output/Analytic/ACTIVE` directory.

Warning: While testing for domain and IP, a subject which has an HTTP code listed below and a global status `INACTIVE` or `INVALID` will get its status updated to `ACTIVE`.

Warning: While testing for URL, if the extracted HTTP code is in the following list, the global status will be `ACTIVE`.

- 100 - Continue
- 101 - Switching Protocols
- 200 - OK
- 201 - Created
- 202 - Accepted
- 203 - Non-Authoritative Information
- 204 - No Content
- 205 - Reset Content
- 206 - Partial Content

18.7.2 As potentially active

Note: While testing for domain and IP, a subject which has an HTTP code listed below, will be saved/logged into the `output/Analytic/POTENTIALLY_ACTIVE` directory.

Warning: While testing for domain and IP, a subject which has an HTTP code listed below and a global status `INACTIVE` or `INVALID` will get its status updated to `ACTIVE`.

Warning: While testing for URL, if the extracted HTTP code is in the following list, the global status will be `ACTIVE`.

- 000

- 300 - Multiple Choices
- 301 - Moved Permanently
- 302 - Found
- 303 - See Other
- 304 - Not Modified
- 305 - Use Proxy
- 307 - Temporary Redirect
- 403 - Forbidden
- 405 - Method Not Allowed
- 406 - Not Acceptable
- 407 - Proxy Authentication Required
- 408 - Request Timeout
- 411 - Length Required
- 413 - Request Entity Too Large
- 417 - Expectation Failed
- 500 - Internal Server Error
- 501 - Not Implemented
- 502 - Bad Gateway
- 503 - Service Unavailable
- 504 - Gateway Timeout
- 505 - HTTP Version Not Supported

18.7.3 As inactive or potentially inactive

Note: While testing for domain and IP, a subject which has an HTTP code listed below, will be saved/logged into the `output/Analytic/POTENTIALLY_INACTIVE` directory.

Warning: While testing for URL, if the extracted HTTP code is in the following list, the global status will be `INACTIVE`.

- 400 - Bad Request
- 401 - Unauthorized
- 402 - Payment Required (Not in use but may be seen in the future)
- 404 - Not Found
- 409 - Conflict
- 410 - Gone
- 412 - Precondition Failed

- 414 - Request-URI Too Long
- 415 - Unsupported Media Type
- 416 - Requested Range Not Satisfiable

18.8 Expiration Date

Give us the expiration date of the tested element.

Note: The API equivalent is `expiration_date`.

18.8.1 Unknown (CLI)

Unknown is returned when we could not extract the expiration date from `PyFunceble.lookup.whois.WhoisLookup.request()` outputs.

18.8.2 None (API)

None is returned when we could not extract the expiration date from `PyFunceble.lookup.whois.WhoisLookup.request()` outputs.

18.8.3 A date

Only if we could extract the expiration date from `PyFunceble.lookup.whois.WhoisLookup.request()`, the date becomes formatted like `02-jan-2017`.

18.9 whois_record (API)

Give us the WHOIS record.

Warning: You may get `DATE EXTRACTED FROM WHOIS DATABASE` as response when it's coming from the JSON database.

The reason behind this is that we don't want to grow the size of the JSON file.

18.10 whois_server (API)

Give us the WHOIS server we contacted in order to get the WHOIS record.

18.11 domain_syntax_validation (API)

The `domain_syntax_validation` provides the response we got while trying to validate (with `PyFunceble.check.Check.is_domain()`) the given subject.

The response will be `True` for the case of a successful validation, `False` otherwise, and `None` for the case that the validation was not executed.

18.12 subdomain_syntax_validation (API)

The `subdomain_syntax_validation` provides the response we got while trying to validate (with `PyFunceble.check.Check.is_subdomain()`) the given subject.

The response will be `True` for the case of a successful validation, `False` otherwise, and `None` for the case that the validation was not executed.

18.13 url_syntax_validation (API)

The `url_syntax_validation` provides the response we got while trying to validate (with `PyFunceble.check.Check.is_url()`) the given subject.

The response will be `True` for the case of a successful validation, `False` otherwise, and `None` for the case that the validation was not executed.

18.14 ipv4_syntax_validation (API)

The `ipv4_syntax_validation` provides the response we got while trying to validate (with `PyFunceble.check.Check.is_ipv4()`) the given subject.

The response will be `True` for the case of a successful validation, `False` otherwise, and `None` for the case that the validation was not executed.

18.15 ipv6_syntax_validation (API)

The `ipv6_syntax_validation` provides the response we got while trying to validate (with `PyFunceble.check.Check.is_ipv6()`) the given subject.

The response will be `True` for the case of a successful validation, `False` otherwise, and `None` for the case that the validation was not executed.

18.16 ipv4_range_syntax_validation (API)

The `ipv4_range_syntax_validation` provides the response we got while trying to validate (with `PyFunceble.check.Check.is_ipv4_range()`) the given subject.

The response will be `True` for the case of a successful validation, `False` otherwise, and `None` for the case that the validation was not executed.

18.17 `ipv6_range_syntax_validation` (API)

The `ipv6_range_syntax_validation` provides the response we got while trying to validate (with `PyFunceble.check.Check.is_ipv6_range()`) the given subject.

The response will be `True` for the case of a successful validation, `False` otherwise, and `None` for the case that the validation was not executed.

19.1 Endpoints

Note: This section document what you can call directly when you use PyFunceble as an imported module.

The tool to check the availability or syntax of domain, IP or URL.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfunceble.github.io/special-thanks.html>

Contributors: <https://pyfunceble.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFunceble>

Project documentation: <https://pyfunceble.readthedocs.io/en/master/>

Project homepage: <https://pyfunceble.github.io/>

License:

Copyright 2017, 2018, 2019, 2020 Nissar Chababy

Licensed under the Apache License, Version 2.0 (the "License");
you may **not** use this file **except in** compliance **with** the License.
You may obtain a copy of the License at

(continues on next page)

(continued from previous page)

```
http://www.apache.org/licenses/LICENSE-2.0
```

Unless required by applicable law **or** agreed to **in** writing, software distributed under the License **is** distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express **or** implied. See the License **for** the specific language governing permissions **and** limitations under the License.

`PyFunceble.dns_lookup(subject, dns_server=None, complete=False, lifetime=3)`

Make a DNS lookup of the given subject.

Parameters

- **subject** (*str*) – The subject we are working with.
- **dns_server** (*str/int*) – A (or list of) DNS server to use while resolving.
- **complete** (*bool*) – Tell us to look for everything instead of NS only.
- **lifetime** (*int*) – The query lifetime.

Returns

A dict with following index if the given subject is not registered into the given DNS server. (More likely local subjects).

```
{
    "hostname": "",
    "aliases": [],
    "ips": []
}
```

A dict with following index for everything else (and if found).

```
{
    "A": [],
    "AAAA": [],
    "CNAME": [],
    "MX": [],
    "NS": [],
    "TXT": [],
    "PTR": []
}
```

Return type `dict`

`PyFunceble.get_complements(subject, include_given=False)`

Provides the complements of the given subject(s).

A complement is for example `example.com` if `www.example.com` is given and vice-versa.

Parameters

- **subject** (*str, list*) – The subject to get the complement for.
- **include_given** (*bool*) – Tell us to add the given one into the result.

Return type `list`

`PyFunceble.ipv4_syntax_check(ip)`

Check the syntax of the given IPv4.

Parameters `ip` (*str/list*) – The IPv4 to check the syntax for.

Returns The syntax validity.

Return type `bool`

Warning: This method will be deprecated one day in the future.

Please report to `is_ipv4()`.

`PyFunceble.is_domain(subject)`

Check if the given subject is a syntactically valid domain.

Parameters `subject` (*str/list*) – The subject to check the syntax from.

Returns The syntax validity.

Return type `bool`

`PyFunceble.is_domain_malicious(subject)`

Checks if the given domain is malicious.

Parameters `subject` (*str*) – The subject to work with.

Return type `bool`

`PyFunceble.is_ip(subject)`

Checks if the given subject is a syntactically valid IPv4 or IPv6.

Parameters `subject` (*str/list*) – The subject to check the syntax from.

Returns The syntax validity.

Return type `bool`

`PyFunceble.is_ip_range(subject)`

Check if the given subject is a syntactically valid IPv4 or IPv6 range.

Parameters `subject` (*str/list*) – The subject to check the syntax from.

Returns The IPv4 range state.

Return type `bool`

`PyFunceble.is_ipv4(subject)`

Check if the given subject is a syntactically valid IPv4.

Parameters `subject` (*str/list*) – The subject to check the syntax from.

Returns The syntax validity.

Return type `bool`

`PyFunceble.is_ipv4_malicious(subject)`

Checks if the given IPv4 is malicious.

Return type `bool`

`PyFunceble.is_ipv4_range(subject)`

Check if the given subject is a syntactically valid IPv4 range.

Parameters `subject` (*str/list*) – The subject to check the syntax from.

Returns The IPv4 range state.

Return type `bool`

`PyFunceble.is_ipv6(subject)`

Checks if the given subject is syntactically valid IPv6.

Parameters `subject` (*str*, *list*) – The subject to check the syntax from.

Returns The syntax validity.

Return type `boolldict`

`PyFunceble.is_subdomain(subject)`

Check if the given subject is a syntactically valid subdomain.

Parameters `subject` (*str/list*) – The subject to check the syntax from.

Returns The syntax validity.

Return type `boolldict`

`PyFunceble.is_url(subject)`

Check if the given subject is a syntactically valid URL.

Parameters `subject` (*str/list*) – The subject to check the syntax from.

Returns The syntax validity.

Return type `boolldict`

`PyFunceble.is_url_malicious(subject)`

Checks if the given URL is malicious.

Parameters `subject` (*str*) – The subject to work with.

Return type `bool`

`PyFunceble.load_config(generate_directory_structure=False, custom=None)`

Load the configuration.

Parameters

- **generate_directory_structure** (*bool*) – Tell us if we generate the directory structure along with loading the configuration file.
- **custom** (*dict*) – A dict with the configuration index (from `.PyFunceble.yaml`) to update.

Note: If `config` is given, the given `dict` overwrite the last value of the given indexes in the configuration.

It's actually something like following:

```
pyfunceble.configuration.update(config_given_by_user)
```

`PyFunceble.syntax_check(domain)`

Check the syntax of the given domain.

Parameters `domain` (*str/list*) – The domain to check the syntax from.

Returns The syntax validity.

Return type `boolldict`

Warning: This method will be deprecated one day in the future.

Please report to `is_domain()`.

PyFunceble.**test** (*subject*, *complete=False*, *config=None*)

Test the availability of the given subject (domain or IP).

Parameters

- **subject** (*str/list*) – The subject (IP or domain) to test.
- **complete** (*bool*) – Activate the return of a dict with some significant data from the test.
- **config** (*dict*) – A dict with the configuration index (from .PyFunceble.yaml) to update.

Returns The status or the informations of the domain.

Return type strldict

Note: If *config* is given, the given dict overwrite the last value of the given indexes in the configuration.

It's actually something like following:

```
pyfunceble.configuration.update(config_given_by_user)
```

Note: If *complete* is set to True, we return the following indexes.

```
{
  "_status_source": None,
  "_status": None,
  "dns_lookup": [],
  "domain_syntax_validation": None,
  "expiration_date": None,
  "http_status_code": None,
  "ipv4_range_syntax_validation": None,
  "ipv4_syntax_validation": None,
  "ipv6_range_syntax_validation": None,
  "ipv6_syntax_validation": None,
  "status": None,
  "status_source": None,
  "subdomain_syntax_validation": None,
  "tested": None,
  "url_syntax_validation": None,
  "whois_record": None,
  "whois_server": None,
}
```

PyFunceble.**url_syntax_check** (*url*)

Check the syntax of the given URL.

Parameters *url* (*str/list*) – The URL to check the syntax for.

Returns The syntax validity.

Return type booldict

Warning: This method will be deprecated one day in the future.

Please report to `is_url()`.

PyFunceble.**url_test** (*subject*, *complete=False*, *config=None*)

Test the availability of the given subject (URL).

Parameters

- **subject** (*str/list*) – The subject (URL) to test.
- **complete** (*bool*) – Activate the return of a dict with some significant data from the test.
- **config** (*dict*) – A dict with the configuration index (from .PyFunceble.yaml) to update.

Returns The status or the informations of the URL.

Return type strldict

Note: If *config* is given, the given dict overwrite the last value of the given indexes in the configuration.

It's actually something like following:

```
pyfunceble.configuration.update(config_given_by_user)
```

Note: If *complete* is set to *True*, we return the following indexes.

```
{
    "_status_source": None,
    "_status": None,
    "dns_lookup": [],
    "domain_syntax_validation": None,
    "expiration_date": None,
    "http_status_code": None,
    "ipv4_range_syntax_validation": None,
    "ipv4_syntax_validation": None,
    "ipv6_range_syntax_validation": None,
    "ipv6_syntax_validation": None,
    "status": None,
    "status_source": None,
    "subdomain_syntax_validation": None,
    "tested": None,
    "url_syntax_validation": None,
    "whois_record": None,
    "whois_server": None,
}
```

PyFunceble.**whois** (*subject*, *server=None*, *timeout=3*)

Request the WHOIS record of the given subject.

Parameters

- **subject** (*str*) – The subject we are working with.
- **server** (*str*) – The WHOIS server to communicate with.

Note: If *None* is given, we look for the best one.

- **timeout** (*int*) – The timeout to apply to the request.

Returns *None* or the WHOIS record.

Return type Nonelstr

19.2 File generation while using the API

You may want to test using the API but still want the result structured normally like a CLI usage. For that case simply add the following.

19.2.1 Global configuration load

```
import PyFunceble

OUR_CUSTOM_PYFUNCEBLE_CONFIGURATION = {"api_file_generation": True}

PyFunceble.load_config(
    generate_directory_structure=True, custom=OUR_CUSTOM_PYFUNCEBLE_CONFIGURATION
)

# The output directory is then generated, and the file(s) too.
# You logic or whatever you want.
```

19.2.2 Local configuration load

```
import PyFunceble

OUR_CUSTOM_PYFUNCEBLE_CONFIGURATION = {"api_file_generation": True}

print(
    "google.com is "
    f"{PyFunceble.test('google.com', config=OUR_CUSTOM_PYFUNCEBLE_CONFIGURATION)}"
)

# The output directory is then generated, and the file(s) too.
```


20.1 AdBlock/Filter list decoding

Warning: This component is not activated by default.

20.1.1 Why do we need it?

As some people may want to test the content of their AdBlock/Filter list, we offer a way to decode them!

20.1.2 How does it work?

Note: Want to read the code ? It's here `PyFunceble.converter.adblock.Adblock.get_converted()`!

We keep it simple by trying to comply with the [Adblock Plus filters explained](#) documentation. For us, the relevant parts are the one which defines/explains which domains are being blocked from a given rule.

Note: A more aggressive extraction might be planned in the future.

20.1.3 How to use it?

You can simply change

```
adblock: False
```

to

```
adblock: True
```

into your personal `.PyFunceble.yaml` or use the `--adblock` argument from the CLI to activate it.

20.2 Autocontinue

20.2.1 Why do we need it?

The auto-continue logic was mainly created for one purpose: Testing long files inside Travis CI. As Travis CI has a time limit of 45 minutes, it became vital for us to be able to stop and continue the test from where we were under those 45 minutes. That's how it started.

Today, - and it might be controversial - it is used by most people who aren't under a Travis CI container to continue when the machine or tool crashes.

20.2.2 How does it work?

Note: Want to read the code ? It's here `PyFunceble.engine.auto_continue.AutoContinue()`!

We log every subject already tested previously and remove them when the same file path is given again.

20.2.3 How to use it?

It is activated by default but if not simply change

```
auto_continue: False
```

to

```
auto_continue: True
```

into your personal `.PyFunceble.yaml` or use the `--auto-continue` argument from the CLI to reactivate it.

20.3 Autosave

Warning: This component is not activated by default.

20.3.1 Why do we need it?

This component comes along with the auto-continue one. Indeed, after constructing the logic to auto-continue we needed something to autosave.

20.3.2 How does it work?

Note: Want to read the code ? It's here `PyFuncible.engine.auto_save.AutoSave()`!

After a given amount of minutes, we stop the tool, generate the percentage, run a given command (if found), commit all the changes we made to the repository and finally, push to the git repository.

20.3.3 How to use it?

For Travis CI and GitLab CI/CD

The following (from the configuration) or their equivalent from the CLI are required.

```
ci: False
ci_autosave_commit: "Your awesome commit message"
ci_autosave_final_commit: "Your awesome final commit message"
ci_autosave_minutes: 15
ci_branch: master
```

Note: If you give the `command_index` something, we will run it at the end of each commits except the last one.

The command on the last commit is executed based on the given `command_before_end` index.

20.4 Certificate verification

Warning: This component is not activated by default.

20.4.1 Why do we need it?

You might sometime be sure that every URL tested with PyFuncible have a valid certificate. This what it's all about!

20.4.2 How does it work?

If the certificate is not valid (catched with `requests`). An `INACTIVE` status is returned (if this component is activated of course)

20.4.3 How to use it?

Simply change

```
verify_ssl_certificate: False
```

to

```
auto_verify_ssl_certificatecontinue: True
```

into your personal `.PyFunceble.yaml` or use the `--verify-ssl-certificate` argument from the CLI to activate it.

20.5 Cleaning

20.5.1 Why do we need it?

Because we constantly need to clean files which are not needed before starting a new test, we embedded our cleaning logic.

20.5.2 How does it work?

Note: Want to read the code ? It's here `PyFunceble.output.clean.Clean()`!

It has an internal map of what has to be deleted and how.

20.5.3 How to clean?

For a simple clean, run PyFunceble with the `--clean` argument.

For a complete cleaning, run PyFunceble with the `--clean-all` argument.

20.5.4 Differences between simple and complete cleaning?

The `--clean` logic cleans the `output/` directory along with the `pyfunceble_tested` table when the `mariadb` or `mysql` database type are choosen.

The `--clean-all` deletes all files generated along with the content of all database tables when the `mariadb` or `mysql` database type are choosen.

Warning: `--clean-all` does not delete the following files (even if generated by us):

- `.pyfunceble-env`
- `whois_db.json`

Warning: `--clean-all` does not delete the content of the following tables:

- `pyfunceble_whois`

20.6 Complements Generation

Warning: This component is not activated by default.

20.6.1 Why do we need it?

Let's say we have `example.org` but `www.example.org` (or vice-versa) is not into my list. This component (if activated) let us test `www.example.org` (or vice-versa) even if it's not into the given list.

20.6.2 How does it work?

Note: Want to read the code ? It's here `PyFunceble.get_complements()` !

At the end of the normal test process, we generate the list of complements and test them.

20.6.3 How to use it?

You can simply change

```
generate_complements: False
```

to

```
generate_complements: True
```

into your personal `.PyFunceble.yaml` or use the `--complements` argument from the CLI to activate it.

20.7 Configuration

20.7.1 Why do we need it?

As we wanted to be hybrid and allow different modes and options, we introduced the configuration logic.

20.7.2 How does it work?

Note: Want to read the configuration loader code ? It's here `PyFunceble.config.load.Load()` !

We first look for the `.PyFunceble.yaml`. If not found, we get/generate it. Then we parse it to the system.

Note: Because we also wanted to get rid of the configuration for an end-user point of view, almost all configuration indexed can be updated from the CLI.

In that case, we update the configuration with the different argument you gives us before parsing it to the system.

Note: If in the future a new configuration key is introduced, you will be asked to choose if you want to merge it into your `.PyFunceble.yaml`.

In that case, we get a copy of the new one and keep/set all previously set indexes. Which means that you don't have to care about reconfiguring previously set indexes.

20.7.3 How to configure?

Update the `.PyFunceble.yaml` file or use the CLI.

20.8 Custom DNS Server

20.8.1 Why do we need it?

Some times the testing environment is setup to use DNS-server which isn't suited for running a PyFunceble test of actually expired or active domains or urls. This could by example be your own DNS-Firewall.

To avoid these situations, the program allows you to setup test DNS-Server.

20.8.2 How does it work?

Thanks to `python-dns` we can parse the given DNS server.

20.8.3 How to use it?

By default, PyFunceble will use the system-wide DNS settings. This can be changed with the ability to configure which DNS-Servers you like PyFunceble to use doing the test.

You set this up with the CLI command `--dns` **or** insert it into your personal `.PyFunceble.yaml`

```
dns_server: null
```

to

```
dns_server:
  - "8.8.8.8"
  - "8.8.8.8"
```

Since `v3.0.0` it is possible to assign a specific port to use with the DNS-Server.

Hint: `--dns 95.216.209.53:53 116.203.32.67:53 9.9.9.9:853`

Warning: If you don't append a port number, the default DNS port (53) will be used.

20.9 Custom User-Agent

20.9.1 Why do we need it?

As we need to be one in a middle of a connection to a webserver, the custom user agent is there for that!

20.9.2 How does it work?

We set the user-agent every time we request something with the `http` and `https` protocols.

If a custom user agent is given, it will be used.

Otherwise, every 24 hours, we update our `user-agents.json` file which will be fetched by your local version to determine the user-agent to use.

20.9.3 How to use it?

Simply choose your browser and platform or provide us your custom one!

```
user_agent:
  browser: chrome
  platform: linux
  custom: null
```

into your personal `.PyFunceble.yaml` or use the `--user-agent` (custom UA) argument from the CLI.

Available Browser

Here is a list of available and accepted browsers at this time.

- `chrome`
- `edge`
- `firefox`
- `ie`
- `opera`
- `safari`

Available Platform

Here is a list of available and accepted platform at this time.

- `linux`
- `macosx`
- `win10`

20.9.4 What if we don't give a custom User-Agent?

If you don't set a custom User-Agent, we will try to get the latest one for the chosen browser and platform.

20.10 Databases

20.10.1 Why do we use “databases”?

We use databases to store data while we run the tests. When globally talking about databases, we are indirectly talking about the following subsystems.

- Autocontinue
- InactiveDB
- Mining
- WhoisDB

Warning: There is a difference between what we are talking here and the `--database` argument which only enable/disable the InactiveDB subsystem.

20.10.2 How do we manage them?

They consist of simple JSON files which are read and updated on the fly.

20.10.3 Warnings around Database (self) management

Warning: If you plan to delete everything and still manage to use PyFunceble in the future, please use the `--clean-all` argument.

Indeed, it will delete everything which is related to what we generated, except things like the whois database file/table which saves (almost) static data which can be reused in the future.

Deleting, for example, the whois database file/table will just make your test run for a much longer time if you retest subject that used to be indexed into the whois database file/table.

20.10.4 Databases types

Since PyFunceble 2.0.0 (equivalent of `>=1.18.0.dev`), we offer multiple database types which are (as per configuration) `json` (default), `mariadb` and `mysql`.

Why different database types?

With the introduction of the multiprocessing logic, it became natural to introduce other database format as it's a nightmare to update a JSON formatted file.

In order to write or use a JSON formatted database, we have to load it and overwrite it completely. It's great while working with a single CPU/process but as soon as we get out of that scope it become unmanageable.

How to use the `mysql` or `mariadb` format?

1. Create a new user, password and database (optional) for PyFunceble to work with.

2. Create a `.pyfunceble-env` file at the root of your configuration directory.
3. Complete it with the following content (example)

```
PYFUNCEBLE_DB_CHARSET=utf8mb4
PYFUNCEBLE_DB_HOST=localhost
PYFUNCEBLE_DB_NAME=PyFunceble
PYFUNCEBLE_DB_PASSWORD=Hello,World!
PYFUNCEBLE_DB_PORT=3306
PYFUNCEBLE_DB_USERNAME=pyfunceble
```

Note: Since version 2.4.3.dev it is possible to use the UNIX socket for the `PYFUNCEBLE_DB_HOST` environment variable.

The typical location for `mysqld.sock` is `/var/run/mysqld/mysqld.sock`.

This have been done to make

1. It easier to use the `socket` in conjunction with a supported CI environment/platform.
2. Leaving more space on the IP-stack on local DB installations.
3. The `UNIX:SOCKET` is usually faster than the IP connection on local runs.

```
PYFUNCEBLE_DB_CHARSET=utf8mb4
PYFUNCEBLE_DB_HOST=/var/run/mysqld/mysqld.sock
PYFUNCEBLE_DB_NAME=PyFunceble
PYFUNCEBLE_DB_PASSWORD=Hello,World!
PYFUNCEBLE_DB_PORT=3306
PYFUNCEBLE_DB_USERNAME=pyfunceble
```

-
4. Switch the `db_type` index of your configuration file to `mysql` or `mariadb`.
 5. Play with PyFunceble!

Note: If the environment variables are not found, you will be asked to prompt the information.

Note: You can find the respective `*.sql` scripts at https://github.com/funilrys/PyFunceble/tree/dev/db_types

20.10.5 Known limitations with MySQL and MariaDB

MySQL and MariaDB's default settings prevent ordinary users from creating stored functions, which can cause unsafe events to be written into the binary log.

You will need to contact your database manager (maybe your self) and ask them to either: set this value (ad: 1) or even better, ask them to import the SQL script directly into the database.

If you are running your own database server, the recommendation is you import the equivalent SQL script directly into the database, as `log_bin` is used for ex. clustering. Turning this value into `1`, might cause security issues later on, as you have lowered the servers default security level of trust.

1. Import the matching SQL script manually as a *SUPER-PRIVILEGED* user such as `root`. **or**
2. Set `set global log_bin_trust_function_creators=1;` for your database.

Warning: We will not recommended this for safety reasons, as it can exposes you for security or stability issues later on, if you run other SQL scripts.

20.10.6 Search string

```
pymysql.err.InternalError: (1419, 'You do not have the SUPER privilege and binary_
↳ logging is enabled (you *might* want to use the less safe log_bin_trust_function_
↳ creators variable)')
```

You do **not** have the SUPER privilege **and** binary logging **is** enabled

20.11 Directory Structure

20.11.1 Why do we need it?

As we wanted the end-user to be able to work from everywhere into the filesystem, we created a logic which will create and keep the `output/` directory which complies with our code.

20.11.2 How does it work?

Note: Want to read the code ? It's here `PyFunceble.output.constructor.Constructor()` !

After each version, the maintainer does a `--production` which will prepare the repository for production. That has the side effect to map the maintainer version of the `output/` directory into a file called `dir_structure_production.json`.

Once pushed, on the end-user side, when testing for file, that file is downloaded into a file called `dir_structure.json` which is then used to restore/create a a perfect copy of the output directory the maintainer had when pushing the new version.

Note: If you find yourself in a case that a directory is not found, please try first to delete the `dir_structure*.json` files to force a resynchronization.

20.11.3 How to generate it manually?

You can't. But using the `--dir-structure` argument will do the job on purpose.

20.12 DNS Lookup

20.12.1 Why do we need it?

As our main purpose is to check the availability of the given subjects, we make a DNS lookup to determine it.

20.12.2 How does it work?

Note: Want to read the code ? It's here `PyFunceble.lookup.dns.DNSLookup.request()` !

For domains

In order:

1. Request the NS record.
2. If not found, request the A record.
3. If not found, request the AAAA record.
4. If not found, request the CNAME record.
5. If not found, request the DNAME record.

Warning: If none is found, we call the UNIX/C equivalent of `getaddrinfo()`.

For IP

We request the PTR record for the IP.

Warning: If none is found, we call the UNIX/C equivalent of `gethostbyaddr()`.

20.13 Environment variables

20.13.1 Dotenv files

Since PyFunceble 2.0.0 (equivalent of PyFunceble-dev >=1.18.0), we load (thanks to [python-dotenv](#)) the content of the following files into the (local) list of environment variables.

1. `.env` (current directory)
2. `.pyfunceble-env` (current directory)
3. `.env` (configuration directory)
4. `.pyfunceble-env` (configuration directory)

To quote the [python-dotenv](#) documentation, a `.env` should look like the following:

```
# a comment and that will be ignored.
REDIS_ADDRESS=localhost:6379
MEANING_OF_LIFE=42
MULTILINE_VAR="hello\nworld"
```

20.13.2 What do we use and why ?

Here is the list of environment variables we use and how we use them if they are set.

Environment Variable	How do we use it?
DEBUG_PYFUNCEBLE	Tell us to log everything into the output/logs/*.log files.
DEBUG_PYFUNCEBLE_ON_SCREEN	Tell us to log everything to stdout
PYFUNCEBLE_AUTO_CONFIG_UPDATE	Tell us if we have to install/update the configuration file automatically.
PYFUNCEBLE_DB_CHARSET	Tell us the MySQL charset to use.
PYFUNCEBLE_DB_HOST	Tell us the host or the Unix socket (absolute file path) of the MySQL database.
PYFUNCEBLE_DB_NAME	Tell us the name of the MySQL database to use.
PYFUNCEBLE_DB_PASSWORD	Tell us the MySQL user password to use.
PYFUNCEBLE_DB_PORT	Tell us the MySQL connection port to use.
PYFUNCEBLE_DB_USERNAME	Tell us the MySQL user-name to use.
PYFUNCEBLE_CONFIG_DIR	Tell us the location of the directory to use as the configuration directory.
PYFUNCEBLE_OUTPUT_DIR	Same as PYFUNCEBLE_CONFIG_DIR it's just present for retro-compatibility.
PYFUNCEBLE_OUTPUT_LOCATION	Tell us where we should generate the output/ directory.
APPDATA	Used under Windows to construct/get the configuration directory if PYFUNCEBLE_CONFIG_DIR is not found.
GH_TOKEN	Tell us the GitHub token to set into the repository configuration when using PyFunceble under Travis CI.
GL_TOKEN	Tell us the GitLab token to set into the repository configuration when using PyFunceble under GitLab CI/CD.
GIT_EMAIL	Tell us the git.email configuration to set when using PyFunceble under any supported CI environment.
GIT_NAME	Tell us the git.name configuration to set when using PyFunceble under any supported CI environment.
TRAVIS_BUILD_DIR	Used to confirm that we are running under a Travis CI container.
GITLAB_CI	Used to confirm that we are running under a GitLab CI/CD environment.
GITLAB_USER_ID	Used to confirm that we are running under a GitLab CI/CD environment.

20.14 Execution time

Warning: This component is not activated by default.

20.14.1 Why do we need it?

As it is always nice to see how long we worked, we added this logic!

20.14.2 How does it work?

Note: Want to read the code ? It's here `PyFunceble.cli.execution_time.ExecutionTime()`!

It shows the exection time on screen (stdout) and at the end of the output/logs/percentage/percentage.txt file if show_percentage is activated.

20.14.3 How to use it?

You can simply change

```
show_execution_time: False
```

to

```
show_execution_time: True
```

into your personal `.PyFunceble.yaml` or use the `--execution` argument from the CLI to activate it.

20.15 List filtering

Warning: This component is not activated by default.

20.15.1 Why do we need it?

While testing for file, you may find yourself in a situation where you only want to test subject which matches a given pattern. That's what this component do.

20.15.2 How does it work?

We scan the list against the given pattern/regex and only test those who match it.

20.15.3 How to use it?

You can simply change

```
filter: ""
```

to

```
filter: "\.org"
```

(for example)

into your personal `.PyFunceble.yaml` or use the `--filter` argument from the CLI.

20.16 IANA Root Zone Database

20.16.1 Why do we need it?

We use it to check if an extension is valid/exists.

20.16.2 How does it work?

Note: Want to read the parser code ? It's here `PyFunceble.lookup.iana.IANA()` !

The root zone database is saved into the `iana-domains-db.json` file. It is formatted like below and is automatically merged for the end-user before each test run.

```
{
    "extension": "whois_server"
}
```

In-app, while testing for a domain, we check if the extension is listed there before doing some extra verifications. If not, domain(s) will be flagged as `INVALID`.

20.16.3 How to generate it manually?

You can't and should not as we are automatically generating it every 24 hours. But using the `--iana` argument will do the job on purpose.

20.17 Test in/for local hostnames, IPs, components

Warning: This component is not activated by default.

20.17.1 Why do we need it?

As we may need to test for local hostnames, IPs, components in a local network, this component allows a less aggressive way of syntax validation.

20.17.2 How does it work?

We simply use a less aggressive syntax validation so that everything you give us is being tested.

20.17.3 How to use it?

Simply change

```
local: False
```

to

```
local: True
```

into your personal `.PyFunceble.yaml` or use the `--local` argument from the CLI to activate it.

20.18 Logs Sharing

Warning: This component is not activated by default.

20.18.1 Why do we need it?

We chose to initiate the logs sharing as some actions can really be random when working with millions of domains. The idea and purpose of this feature are **ONLY** to make PyFunceble a better tool.

20.18.2 What do we share/collect?

Event	Shared	URL
No WHOIS server (referrer) is found.	The extension of the currently tested domain.	<code>https://pyfunceble.funilrys.com/api/no-referer</code>
The expiration date is not correctly formatted.	- The extracted expiration date. - The currently tested domain. - The currently used WHOIS server (DNS) name.	<code>https://pyfunceble.funilrys.com/api/date-format</code>

20.18.3 How to share logs?

If you wish to share your logs simply change

```
share_logs: False
```

to

```
share_logs: True
```

into your personal `.PyFunceble.yaml` or use the `--share-logs` argument from the CLI to activate it.

20.19 Mining

Warning: This component is not activated by default.

20.19.1 Why do we need it?

Sometimes you might, for example, want to get the list of domain(s) / URL(s) in a redirecting loop. This feature reveals them.

20.19.2 How does it work?

Note: Want to read the code ? It's here `PyFunceble.engine.mining.Mining()`!

We access the given domain/URL and get the redirection history which we then test once we finished the normal test.

Note: This component might evolve with time.

20.19.3 How to use it?

You can simply change

```
mining: False
```

to

```
mining: True
```

into your personal `.PyFunceble.yaml` or use the `--mining` argument from the CLI to activate it.

20.20 Multiprocessing

Warning: This component is not activated by default.

20.20.1 Why do we need it?

Many people around the web who talked about PyFunceble were talking about one thing: We take time to run.

This component allows you to use more than one process if your machine has multiple CPU.

Note: If you use this component you have to consider some limits:

- Your connection speed.
- Your machine.

You might not even see a speed if one of both is slow or very slow.

The following might not be touched by the limits but it really depends:

- URL availability test.
 - Syntax test.
 - Test with DNS LOOKUP only - without WHOIS.
-

20.20.2 How does it work?

We test multiple subjects at the same time over several processes (1 process = 1 subject tested) and generate our results normally.

Note: While using the JSON format for the database you might have to wait a bit at the very end as we need to merge all data we generated across the past created processes.

Therefore, we recommend using the MySQL/MariaDB format which will get rid of that as everything is saved/synchronized at an almost real-time scale.

20.20.3 How to use it?

Activation

You can simply change

```
multiprocess: False
```

to

```
multiprocess: True
```

Number of processes to create

Simply update the default value of

```
maximal_processes: 25
```

Warning: If you do not explicitly set the `--processes` argument, we overwrite the default to the number of available CPU.

Merging mode

2 merging cross process (data) merging mode are available:

- `end`
- `live`

With the `end` mode, we merge all data at the very end of the current instance. With the `live` mode, we merge all data while testing.

Simply update the default value of

```
multiprocess_merging_mode: end
```

to the mode you want.

20.21 Outputed Files

Note: This section does not cover the logs files.

20.21.1 Why do we need it?

We need a way to deliver our results.

20.21.2 How does it work?

After testing a given subject, we generate its output file based on what's needed.

Host format

This is the default output file.

A line is formatted like `0.0.0.0 example.org`.

Note: A custom IP can be set with the help of the `custom_ip` index or the `-ip` argument from the CLI.

Don't need it? Simply change

```
generate_hosts: True
```

to

```
generate_hosts: False
```

into your personal `.PyFunceble.yaml` or use the `--hosts` argument from the CLI to deactivate it.

Plain format

A line is formatted like `example.org`. . Need it? Simply change

```
plain_list_domain: False
```

to

```
plain_list_domain: True
```

into your personal `.PyFunceble.yaml` or use the `--plain` argument from the CLI to activate it.

JSON format

Need it? Simply change

```
generate_json: False
```

to


```
generate_json: True
```

into your personal `.PyFunceble.yaml` or use the `--json` argument from the CLI to activate it.

20.22 Percentage

Warning: This component is activated by default while testing files.

Note: The percentage doesn't show up - by design - while testing for single domains (whilst using `--domain`).

20.22.1 Why do we need it?

We need it in order to get information about the amount of data we just tested.

20.22.2 How does it work?

Note: Want to read the code ? It's here `PyFunceble.output.percentage.Percentage()` !

Regularly or at the very end of a test we get the number of subjects for each status along with the number of tested subjects. We then generate and print the percentage calculation on the screen (`stdout`) and into `output/logs/percentage/percentage.txt`

20.22.3 How to use it?

It is activated by default, but if not please update

```
show_percentage: False
```

to

```
show_percentage: True
```

into your personal `.PyFunceble.yaml` or use the `--percentage` argument from the CLI to reactivate it.

20.23 The Public Suffix List

20.23.1 Why do we need it?

We use it in the process of checking the validity of domains.

20.23.2 How does it work?

Note: Want to read the parser code ? It's here `PyFunceble.lookup.publicsuffix.PublicSuffix()` !

The copy of the public suffix list we use is saved into the `public-suffix.json` file. It is formatted like below and is automatically merged for the end-user before each test run.

```
{
  "extension": [
    "suffix1.extension",
    "suffix2.extension",
    "suffix3.extension"
  ]
}
```

In-app, while testing for domain(s), we use it in order to know if we are checking for a subdomain or not.

20.23.3 How to generate it manually?

You can't and should not as we are automatically generating it every 24 hours. But, using the `--public-suffix` argument will do the job on purpose.

20.24 Sorting

Note: While using the multiprocessing option, the data are tested as given.

20.24.1 Why do we need it?

Because sorted is better, we sort by default!

20.24.2 How does it work?

Note: Want to read the code ? It's here: `Sort()` !

Alphabetically

This is the default one. The default python `sorted()` function is used for that purpose.

Hierarchically

The objective of this is to provide sorting by service/domains.

Note: This is a simplified version of what we do.

1. Let's say we have `aaa.bbb.ccc.tdl`.

Note: The TDL part is determined. Indeed we first look at the IANA Root Zone database, then at the Public Suffix List.

2. Let's split the points. We then get a list `[aaa, bbb, ccc, tdl]`
3. Put the TDL first. It will gives us `[tdl, aaa, bbb, ccc]`
4. Reverse everything after the TDL. It will gives us `[tdl, ccc, bbb, aaa]`.
5. Get the string to use for sorting. It will gives us `tdl.ccc.bbb.aaa`.

20.24.3 How to activate the hierarchical sorting?

Simply change

```
hierarchical_sorting: False
```

to

```
hierarchical_sorting: True
```

into your personal `.PyFunceble.yaml` or use the `--hierarchical` argument from the CLI to activate it.

20.25 Whois Lookup

Note: While testing using PyFunceble, subdomains, IPv4 and IPv6 are not used against our whois lookup logic.

20.25.1 Why do we need it?

As our main purpose is to check the availability of the given subjects, we make a WHOIS lookup (if authorized) to determine it.

20.25.2 How does it work?

Note: Want to read the code ? It's here `PyFunceble.lookup.whois.WhoisLookup()` !

For us the only relevant part is the extraction of the expiration date. Indeed, it's an indicator if a domains is still owned by someone, we use it first to get the availability of domains.

20.25.3 How to use it?

It is activated by default but if not simply change

```
no_whois: True
```

to

```
no_whois: False
```

into your personal `.PyFunceble.yaml` or use the `--no-whois` argument from the CLI to reactivate it.

21.1 Submit changes

Before anything, please keep in mind the following. If one or more of those conditions are not filled. Your Pull Request to PyFunceble may not be merged.

The `master` branch is used only for releasing a new or stable version of the code. That's why we require that all contributions/modifications must be done under the `dev` or a new branch.

In order to gain some time and also understand what you are working on, your pull requests submission as long as your commit descriptions should be clear and complete as much as possible. We do an exception to commit with minor changed but big changes should have a complete description. Please ensure to use the following method when committing a big change.

21.2 Coding conventions

21.2.1 Format

1. We should format our code with `Black`, *The uncompromising Python code formatter*.
2. We should write **docstrings** for every public method, function and classes. It does not have to be elaborate. Just explain what it simply does!

```
$ black PyFunceble && black tests
```

21.2.2 Linting

Our code should pass a `pylint` check without any issue (exit 0).

```
$ pylint PyFunceble && pylint tests
```

21.2.3 Coverage

Our code should have at least 60% of coverage.

Note: Coverage irrelevant code/files can be muted.

```
$ coverage run setup.py test && coverage report -m
```

21.2.4 Distribution

Our code should be shipped (after each commit) with a new version and a new `version.yaml` file. That file should **ALWAYS** be generated with the following.

Note: An exception is granted for detached branch (not dev nor master) and no code changes.

```
$ # Prepare our files, :code:`version.yaml` and code for production/dev usage.
$ PyFunceble --production
```

21.2.5 Commit

All your commits should be - when possible - be signed with **PGP**. (More information can be found on [GitHub documentation](#)) Please note the usage of `-S` into the commit command which means that we sign the commit.

The commit message may be formatted like follow:

Note: An exception is granted for no code changes.

:: Short explanation (max 79 char).

Paragraphs with some details/context (if needed).

21.2.6 Git Hook

If one may want to automate the process.

Here is a git hook (as proposition) to be set into `.git/hooks/pre-commit`.

```
#!/usr/bin/env bash

# We print what's going on.
set -x
# We stop on error.
set -e

# We get top level path.
rootDirectory="$(git rev-parse --show-toplevel)"
# We construct our list of directories.
ourDirectories="${rootDirectory}/PyFunceble ${rootDirectory}/tests"
```

(continues on next page)

(continued from previous page)

```
if [[ ! -z "$(git status --porcelain | awk '{ if ($NF > 0 && substr($1,1,1) != "?" &&
↪$2 ~ '/\.py/' ) print $2}')" ]]
then
    hash find
    hash isort
    hash black
    hash pylint
    hash coverage

    for directory in $(echo ${ourDirectories})
    do
        # We sort the imports.
        find "${directory}" -name "*.py" -exec isort {} \;
        # We format the code.
        black "${directory}"
        # We lint the code.
        pylint "${directory}"
    done

    cd "${rootDirectory}"
    coverage run setup.py test
fi

set +e
set +x
exit 0
```

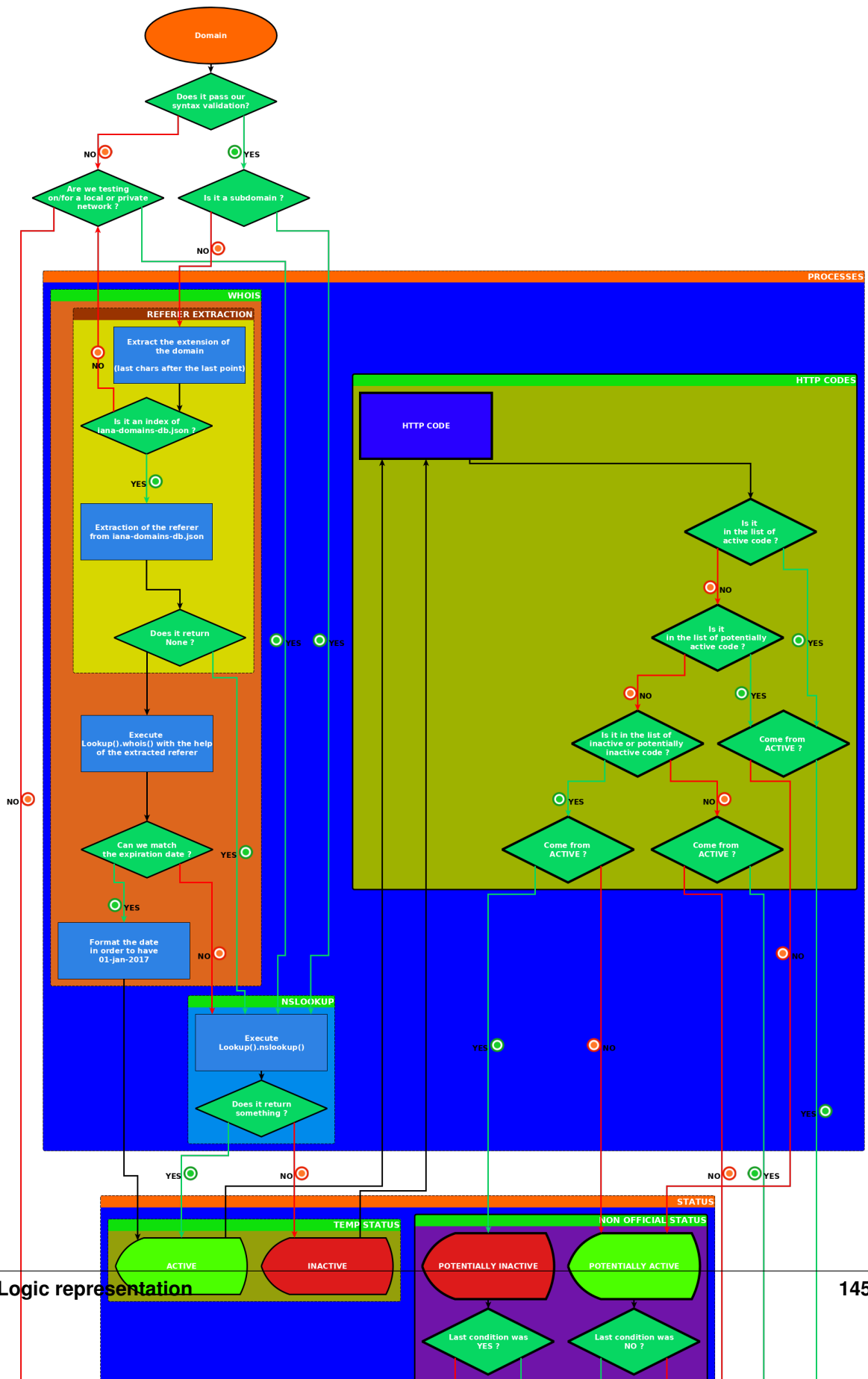

22.1 Logic representation

Don't have time to review the code and want to know exactly what PyFuncible does in general? This flowchart is there for you!

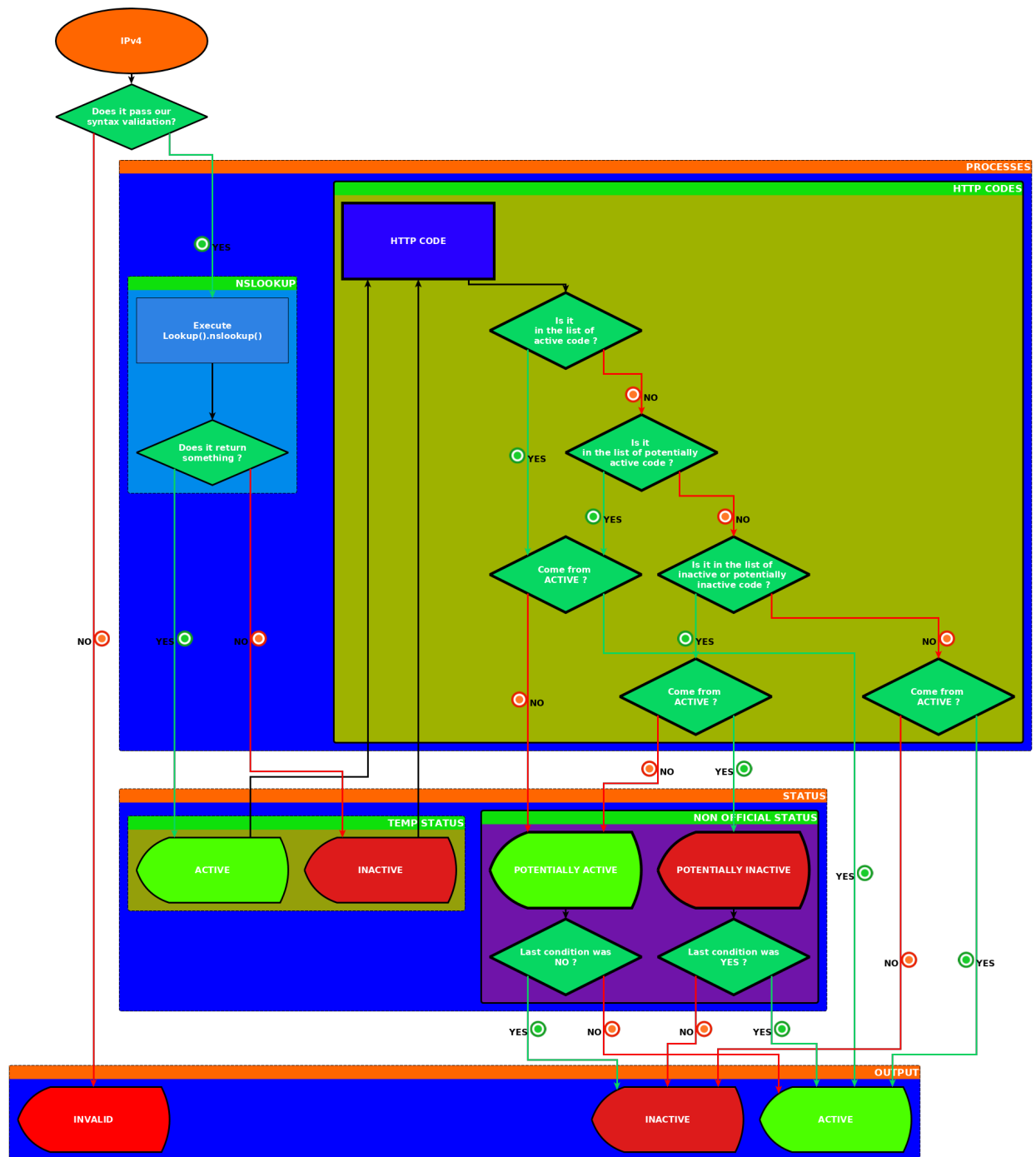
Note: This is a simplified version of what we do.

Note: The diagrams were generated with the help of [yEd Graph Editor](#).

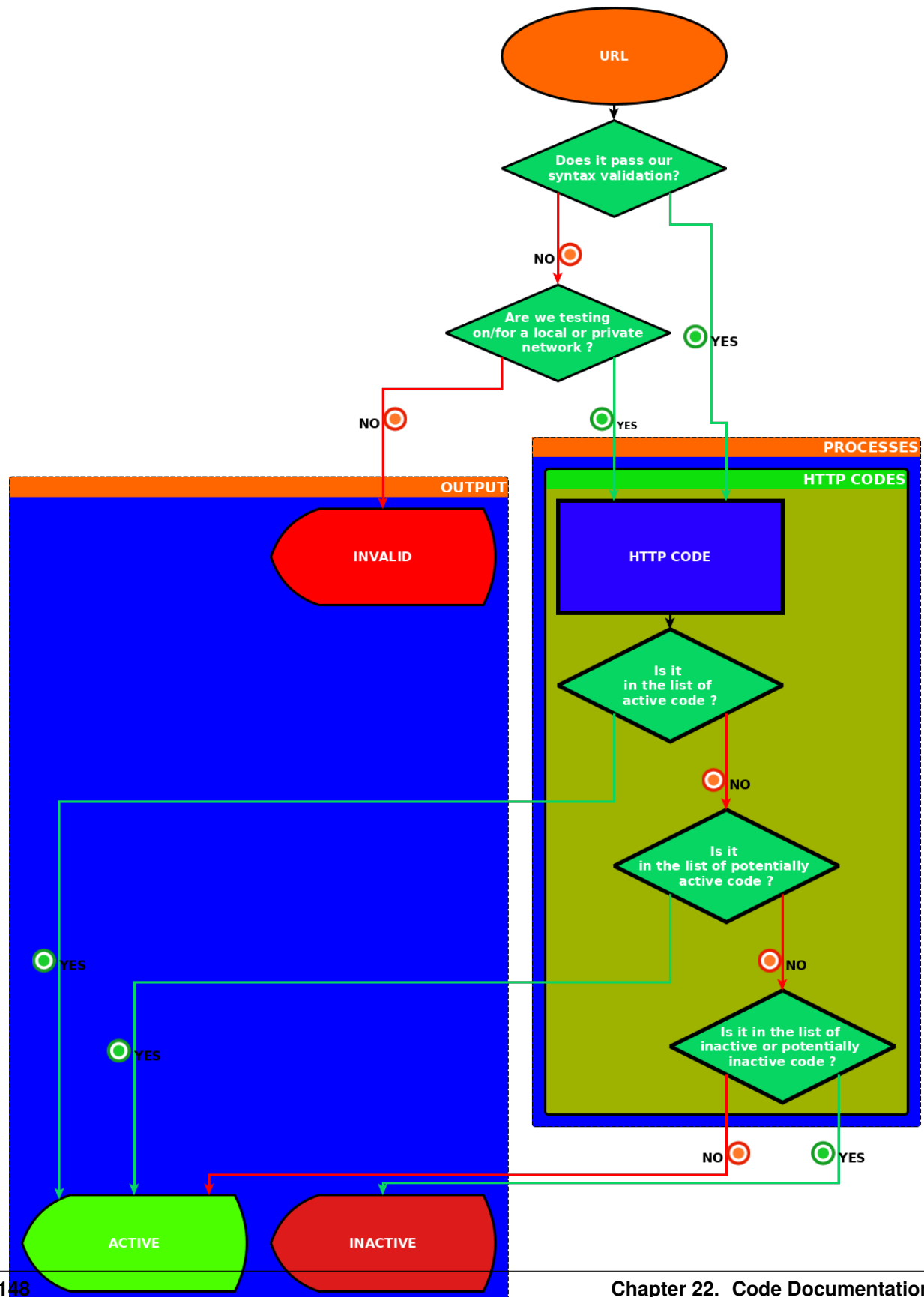
22.1.1 Domain testing



22.1.2 IP testing



22.1.3 URL testing



22.2 Check

22.2.1 Problematic

How can we efficiently check the syntax of IP, domains, and URL?

22.2.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.

Provides the syntax checker interface.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfunceble.github.io/special-thanks.html>

Contributors: <https://pyfunceble.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFunceble>

Project documentation: <https://pyfunceble.readthedocs.io/en/master/>

Project homepage: <https://pyfunceble.github.io/>

License:

```
Copyright 2017, 2018, 2019, 2020 Nissar Chababy
```

```
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
```

```
http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

class `PyFunceble.check.Check` (*subject*)

Provides our syntax checkers.

Parameters `subject` (*str*) – The subject (URL, IP or domain) to check.

SPECIAL_USE_DOMAIN_NAMES_EXTENSIONS = `['onion']`

Specifies the extension which are specified as “Special-Use Domain Names” and supported by our project.

Type `list`

See also:

- [RFC6761](#)

- [IANA Special-Use Domain Names](#) assignments.
- [RFC7686](#)

is_domain (*subdomain_check=False*)

Checks if the given subject is a valid domain.

Parameters **subdomain_check** (*bool*) – Activates the subdomain checking.

Warning: Do not manually use.
Please report to `is_subdomain()`.

Returns The validity.

Return type `bool`

is_ip ()

Checks if the given subject is a valid IPv4 or IPv6.

Returns The validity.

Return type `bool`

is_ip_range ()

Checks if the given subject is a valid IPv4 or IPv6 range.

Returns The validity.

Return type `bool`

is_ipv4 ()

Checks if the given subject is a valid IPv4.

Returns The validity.

Return type `bool`

is_ipv4_range ()

Checks if the given subject is a valid IPv4 range.

Returns The validity.

Return type `bool`

is_ipv6 ()

Checks if the given subject is a valid IPv6.

Returns The validity.

Return type `bool`

is_ipv6_range ()

Checks if the given subject is a valid IPv6 range.

Returns The validity.

Return type `bool`

is_reserved_ip ()

Checks if the given subject is a reserved IPv4 or IPv6.

Returns The validity.

Return type `bool`

is_reserved_ipv4()

Checks if the given subject is a reserved IPv4.

Note: This method has been written on basis of the following links:

- https://en.wikipedia.org/wiki/Reserved_IP_addresses
 - <https://www.iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-registry.xhtml>
-

Returns The validity.

Return type `bool`

is_reserved_ipv6()

Checks if the given subject is a reserved IPv6.

Returns The validity.

Return type `bool`

is_subdomain()

Checks if the given subject is a valid subdomain.

Returns The validity.

Return type `bool`

is_url(return_base=False, return_formatted=False)

Checks if the given subject is a valid URL.

Parameters

- **return_base** (`bool`) – Allow us the return of the url base (if URL formatted correctly).
- **return_formatted** (`bool`) – Allow us to get the formatted URL as response.

Returns The validity or the base if asked.

Return type `bool, str`

22.3 Exceptions

22.3.1 Problematic

How can we provide our very own exceptions ?

22.3.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.



Provides our exceptions.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfunceble.github.io/special-thanks.html>

Contributors: <https://pyfunceble.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFunceble>

Project documentation: <https://pyfunceble.readthedocs.io/en/master/>

Project homepage: <https://pyfunceble.github.io/>

License:

```
Copyright 2017, 2018, 2019, 2020 Nissar Chababy
```

```
Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at
```

```
http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.
```

exception `PyFunceble.exceptions.ConfigurationFileNotFound`
Describes a missing configuration file.

exception `PyFunceble.exceptions.GitEmailNotFound`
Describes a missing Git Email.

exception `PyFunceble.exceptions.GitHubTokenNotFound`
Describes a missing GitHub token.

exception `PyFunceble.exceptions.GitLabTokenNotFound`
Describes a missing GitLab token.

exception `PyFunceble.exceptions.GitNameNotFound`
Describes a missing Git Name.

exception `PyFunceble.exceptions.NoConversionMade`
Describes the fact that a conversion was expected but none was made.

exception `PyFunceble.exceptions.NoDownloadDestinationGiven`
Describes the fact that the download destination was not declared.

exception `PyFunceble.exceptions.NoDownloadLinkGiven`
Describes the fact that no download link was declared.

exception `PyFunceble.exceptions.NoExtractionMade`
Describes the fact that an extraction was expected but none was made.

exception `PyFunceble.exceptions.NoInternetConnection`
Describes a missing connection.

exception `PyFunceble.exceptions.PleaseUpdatePyFunceble`
Describes the impossibility to continue with an older version.

exception `PyFunceble.exceptions.PyFuncebleException`
Describes our own exceptions.

exception `PyFunceble.exceptions.PyFuncebleExternalException`
Describes an exception which is caused by an external input.

exception `PyFunceble.exceptions.PyFuncebleInternalException`
Describes an exception which is caused by our own logic.

exception `PyFunceble.exceptions.UnknownSubject`
Describes the fact that an unknown subject is inputed.

exception `PyFunceble.exceptions.UserAgentBrowserNotFound`
Describes the fact that we could not find a valid browser to work with.

exception `PyFunceble.exceptions.UserAgentNotFound`
Describes the fact that we could not find a user agent to work with.

exception `PyFunceble.exceptions.UserAgentPlatformNotFound`
Describes the fact that we could not find a valid browser to work with.

exception `PyFunceble.exceptions.WrongParameterType`
Describes a wrong parameter type.

exception `PyFunceble.exceptions.WrongParameterValue`
Describes a wrong parameter value.

22.4 Abstracts

22.4.1 Problematic

How can we work with abstracted data and information which are not directly related to our project main logic?

22.4.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.



Provides the abstracted logics/data.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfunceble.github.io/special-thanks.html>

Contributors: <https://pyfunceble.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFunceble>

Project documentation: <https://pyfunceble.readthedocs.io/en/master/>

Project homepage: <https://pyfunceble.github.io/>

License:

Copyright 2017, 2018, 2019, 2020 Nissar Chababy

Licensed under the Apache License, Version 2.0 (the "License");
you may **not** use this file **except in** compliance **with** the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law **or** agreed to **in** writing, software
distributed under the License **is** distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express **or** implied.
See the License **for** the specific language governing permissions **and**
limitations under the License.

Infrastructure()

class PyFunceble.abstracts.infrastructure.Infrastructure

Provides some infrastructure related abstractions.

CONFIGURATION_FILENAME = '.PyFunceble.yaml'

Sets the name of the user editable configuration file.

Type str

DEFAULT_CONFIGURATION_FILENAME = '.PyFunceble_production.yaml'

Sets the name of the default configuration file.

Type str

DOWN_FILENAME = '.pyfunceble_intern_downtime.json'

Sets the name of our own download time record file.

Type str

ENV_FILENAME = '.pyfunceble-env'

Sets the name of our own environment file.

Type str

IANA_FILENAME = 'iana-domains-db.json'

Sets the name of our own IANA database file.

Type str

IPV4_REPUTATION_FILENAME = 'ipv4_reputation.data'

Sets the name of our own copy of the IPv4 reputation data file.

Type str

PROD_CONFIG_LINK = 'https://raw.githubusercontent.com/funilrys/PyFunceble/master/.PyFunceble_production.yaml'

Sets the link to the production configuration file.

Type str

REPO_LINK = 'https://git.io/vpZoI'

Sets the link to the repository.

Type str

USER_AGENT_FILENAME = 'user_agents.json'

Sets the name of our own copy of the user agents file.

Package()

```
class PyFunceble.abstracts.package.Package
    Provides some packaging related abstractions.

    NAME = 'PyFunceble'
        Sets the package name.

        Type str

    VERSION = '3.1.20.dev (Teal Blauwbok: Moth)'
        Sets the package version.

        Type str
```

Platform()

```
class PyFunceble.abstracts.platform.Platform
    Provides an easy way to get the current platform.

    MAC = ['darwin']
        Provides the list of supported MAC platform.

        Type list

    UNIX = ['linux', 'darwin']
        Provides the list of supported unix platform.

        Type list

    WINDOWS = ['windows', 'cygwin', 'cygwin_nt-10.0']
        Provides the list of supported windows platform.

        Type list

    classmethod get()
        Returns the current platform.

        Return type str

    classmethod is_mac_os()
        Checks if the current platform is in our OSX list.

        Return type bool

    classmethod is_unix()
        Checks if the current platform is in our unix list.

        Return type bool

    classmethod is_windows()
        Checks if the current platform is in our windows list.

        Return type bool
```

Version()

```
class PyFunceble.abstracts.package.Version
    Provides a simple way to compare our own versions.

    classmethod compare (upstream)
        Compares the given versions with the local one.
```

Parameters

- **local** (*list*) – The local version converted by `split_versions()`.
- **upstream** (*list*) – The upstream version converted by `split_versions()`.

Returns

- `True`: `local < upstream`
- `None`: `local == upstream`
- `False`: `local > upstream`

Return type `bool, None`**classmethod** `is_local_cloned()`Checks if the local version is was downloaded per `git clone`.**classmethod** `is_local_dev()`

Checks if the local version is the development version.

classmethod `literally_compare(local, upstream)`

Compares the given versions, literally.

Parameters

- **local** (*str*) – The local version converted by `split_versions()`.
- **upstream** (*str*) – The upstream version converted by `split_versions()`.

Returns

- `True`: `local == upstream`
- `False`: `local != upstream`

Return type `bool`**classmethod** `split_versions(version, return_non_digits=False)`

Converts the versions to a shorter one.

Parameters

- **version** (*str*) – The version to split.
- **return_non_digits** (*bool*) – Activate the return of the non-digits parts of the split-version.

Returns A tuple. The first index is the digit part of the version, when the second one is the the non-digit part of the version.**Return type** `tuple`

22.5 CLI

22.5.1 Problematic

How can we easily work and distribute some CLI related logic?

22.5.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.

Provides the CLI entry points.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfunceble.github.io/special-thanks.html>

Contributors: <https://pyfunceble.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFunceble>

Project documentation: <https://pyfunceble.readthedocs.io/en/master/>

Project homepage: <https://pyfunceble.github.io/>

License:

Copyright 2017, 2018, 2019, 2020 Nissar Chababy

Licensed under the Apache License, Version 2.0 (the "License");
you may **not** use this file **except in** compliance **with** the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law **or** agreed to **in** writing, software
distributed under the License **is** distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express **or** implied.
See the License **for** the specific language governing permissions **and**
limitations under the License.

`PyFunceble.cli.tool()`

Provides the CLI.

Dispatcher()

```
class PyFunceble.cli.dispatcher.Dispatcher(preset, domain_or_ip=None, file_path=None,
                                             url_file_path=None, url_to_test=None,
                                             generate_results_only=False, gener-
                                             ate_all_results_only=False)
```

Dispatches to one of the other brain.

Note: We are talking about brain because, each `PyFunceble.core.*` is a unique brain which has it's own function ;-)

Sometimes, 2 brains depends from the other one. But that's great because working together is always great ;-)

Parameters

- **preset** (*Preset*) – An instance of the configuration preset.
- **domain_or_ip** (*list*) – A list of subject to test.
- **file_path** (*str*) – A file path or a link.

Note: If a file path is given, we read and test its content.

Note: If a link is given, we download it's content and test it.

Warning: If given, we consider each line to be an adblock filter, a host file format or a subject to test.

- **url_file_path** (*str*) – A file path or a link.

Note: If a file path is given, we read and test its content.

Note: If a link is given, we download it's content and test it.

Warning: If given, we consider each line to be a URL to test.

- **url_to_test** (*str*) – A list of URL to test.

classmethod dispatch_file_test (*file_path*, *generate_results_only*, *generate_all_results_only*)

Dispatches to the brain in charge of file testing.

Parameters

- **file_path** (*str*) – The file path to test.
- **generate_results_only** (*bool*) – Tell us to only regenerate from the data stored into the MariaDB/MySQL databases.
- **generate_all_results_only** (*bool*) – Tell us to only regenerate from the data stored into the MariaDB/MySQL databases.

Note: The difference with `generate_results_only` is that it includes the retested which status didn't changed.

dispatch_url_file_test (*url_file_path*, *generate_results_only*, *generate_all_results_only*)

Dispatches to the brain in charge of url file testing.

Parameters

- **url_file_path** (*str*) – The file to test.
- **generate_results_only** (*bool*) – Tell us to only regenerate from the data stored into the MariaDB/MySQL databases.

ExecutionTime()

class PyFunceble.cli.execution_time.**ExecutionTime** (*action='start', last=False*)
 Sets and returns the execution time of the program.

Parameters

- **action** (*str*) – The action related the execution time. Can be *start* or *stop*.
- **last** (*bool*) – Tell the subsystem if we are at the very end of the file testing.

classmethod authorization ()

Checks the execution authorization.

Returns The authorization status.

Return type *bool*

classmethod calculate (*start=None, end=None*)

Calculates the difference between starting and ending time.

Parameters

- **start** (*int, str*) – A starting time.
- **stop** (*int, str*) – A ending time.

Returns

A dict with following as index.

- *days*
- *hours*
- *minutes*
- *seconds*

as index.

Return type *dict*

format_execution_time (*start=None, end=None*)

Formats the calculated time into a human readable format.

Parameters

- **start** (*int, str*) – A starting time.
- **stop** (*int, str*) – A ending time.

Returns A human readable date.

Return type *str*

save (*last=False*)

Saves the current time to our backup file.

Parameters **last** (*bool*) – Tell us if we are at the very end of the file testing.

classmethod set_starting_time ()

Sets the starting time.

classmethod set_stoping_time ()

Sets the ending time.

22.6 Config

22.6.1 Problematic

- How can we make personalization more simple?
- How can we automatically merge the new configuration with the local one?
- How can we link some configuration endpoints with the real logic.

22.6.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.



Provides the configuration logics.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfunceble.github.io/special-thanks.html>

Contributors: <https://pyfunceble.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFunceble>

Project documentation: <https://pyfunceble.readthedocs.io/en/master/>

Project homepage: <https://pyfunceble.github.io/>

License:

```
Copyright 2017, 2018, 2019, 2020 Nissar Chababy
```

```
Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at
```

```
http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.
```

Load()

Merge()

Preset()

```
class PyFunceble.config.preset.Preset
    Checks or update the global configuration based on some events.

    classmethod _Preset_are_we_allowed_to_overwrite (index)
        Checks if we are allowed to overwrite an index.

    api ()
        Prepares the global configuration for a test from the API.

    complements ()
        Prepares the global configuration for a complements generation.

    cooldown_time ()
        Ensures that we always have a correct cooldown time.

    classmethod db_types ()
        Ensure that the files are downloaded when the db types is not the JSON one.

    classmethod disable (indexes)
        Sets the given configuration index to False.

    classmethod dns_lookup_over_tcp ()
        Ensures that the DNS lookup over tcp is properly set.

    classmethod dns_nameserver ()
        Ensures that the DNS nameserver is properly set.

    classmethod enable (indexes)
        Sets the given configuration index to True.

    file_url ()
        Prepares the global configuration for a list of URL to test.

    init_all ()
        Initiate all presets which are independent from others.

    classmethod maximal_processes ()
        Ensures that the number of maximal processes is always  $\geq 1$ .

    multiprocess ()
        Prepares the global configuration for a test with multiple processes.

    classmethod multiprocess_merging_mode ()
        Ensures that a valid merging mode is given.

    reputation_data ()
        Ensures that the usage of reputation data is activated when needed.

    classmethod reset_counters ()
        Resets the counters.

    simple_domain ()
        Prepares the global configuration for a domain test.

    simple_url ()
        Prepares the global configuration for an URL test.

    classmethod switch (variable, custom=False)
        Switches PyFunceble.CONFIGURATION variables to their opposite.
```

Parameters

- **variable** (*str/bool*) – The variable name to switch. The variable should be an index our configuration system. If we want to switch a bool variable, we should parse it here.
- **custom** (*bool*) – Let us know if have to switch the parsed variable instead of our configuration index.

Returns The opposite of the configuration index or the given variable.

Return type `bool`

Raises

Exception When the configuration is not valid. In other words, if the `PyFunceble.CONFIGURATION[variable_name]` is not a bool.

syntax_test ()

Disables the HTTP status code if we are testing for syntax

classmethod timeout ()

Ensures that the timeout is always correct.

22.7 Converter

22.7.1 Problematic

How can we convert some data into something else or something our system understand.

22.7.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.

Provides the converters.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfunceble.github.io/special-thanks.html>

Contributors: <https://pyfunceble.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFunceble>

Project documentation: <https://pyfunceble.readthedocs.io/en/master/>

Project homepage: <https://pyfunceble.github.io/>

License:

Copyright 2017, 2018, 2019, 2020 Nissar Chababy

Licensed under the Apache License, Version 2.0 (the "License");
you may **not** use this file **except in** compliance **with** the License.
You may obtain a copy of the License at

(continues on next page)

(continued from previous page)

```
http://www.apache.org/licenses/LICENSE-2.0
```

Unless required by applicable law **or** agreed to **in** writing, software distributed under the License **is** distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express **or** implied. See the License **for** the specific language governing permissions **and** limitations under the License.

ConverterBase()

class PyFunceble.converter.base.**ConverterBase**(*data_to_convert*)
Provides the converter base.

Parameters *data_to_convert* – The data to convert.

get_converted()
Provides the converted data.

AdBlock()

class PyFunceble.converter.adblock.**AdBlock**(*data_to_convert*, *aggressive=False*)
Converts an adblock filter line to a list of subject to test.

__AdBlock__decode_v1(*data*)
Decodes the v1.
Parameters *data* (*str*) – A string to decode.

Return type *list*

__AdBlock__decode_v2(*data*)
Decodes the v2.
Parameters *data* (*str*) – A string to decode.

Return type *list*

__AdBlock__decode_v3(*data*)
Decodes the v3.
Parameters *data* (*str*) – A string to decode.

Return type *list*

__AdBlock__decode_v4(*data*)
Decodes the v4.
Parameters *data* (*str*) – A string to decode.

Return type *list*

extract_base(*subject*)
Extracts the base of the given element.

As an example: given "hello.world/?is=beautiful" returns "hello.world"

Parameters *subject* – The subject to work with.

extract_from_options (*options*)

Extracts the relevant data from the list of options.

Parameters **options** (*list*) – The list of options of a rule.

Returns The list of domains.

Return type *list*

format_decoded (*decoded*, *result=None*)

Formats the extracted adblock line in order to be compatible with what the system understand.

Parameters

- **decoded** (*str*) – The decoded data to work with.
- **result** (*list*) – A list of the result of this method.

Returns The list of domains or IP compatible with the system.

Return type *list*

get_converted ()

Converts and return the result of the conversion.

Return type *list*

classmethod ignore_it (*subject*)

Checks if we have to ignore the given subject.

Parameters **subject** (*str*) – The subject ot work with.

Returns The result of the check.

Return type *bool*

remove_ignored (*subject*)

Removes the ignored element from the given list of subject.

Digit2Digits ()

class PyFunceble.converter.digit2digits.**Digit2Digits** (*data_to_convert*)

Converts a given digit to a 2 digits string.

File ()

class PyFunceble.converter.file.**File** (*data_to_convert*)

Converts a line to a subject to test.

get_converted ()

Provides the converted data.

Warning: This method returns return None if no subject of interest was found.

Return type *None, str, list*

`InternalUrl()`

class `PyFuncible.converter.internal_url.InternalUrl(data_to_convert)`
 Converter of the internal URLs.

Note: The internal URLs are actually the URL that has nothing to do with what we are going to test.
 They are only relevant for the software itself.

Parameters `data_to_convert` (*str*) – The data to convert

`to_right_url()`
 Process the conversion to the right URL.

22.8 Core

22.8.1 Problematic

How can we get started?

22.8.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.

Provides the core interfaces.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfuncible.github.io/special-thanks.html>

Contributors: <https://pyfuncible.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFuncible>

Project documentation: <https://pyfuncible.readthedocs.io/en/master/>

Project homepage: <https://pyfuncible.github.io/>

License:

Copyright 2017, 2018, 2019, 2020 Nissar Chababy

Licensed under the Apache License, Version 2.0 (the "License");
 you may **not** use this file **except in** compliance **with** the License.
 You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

(continues on next page)

(continued from previous page)

Unless required by applicable law **or** agreed to **in** writing, software distributed under the License **is** distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express **or** implied. See the License **for** the specific language governing permissions **and** limitations under the License.

APICore()

```
class PyFuncible.core.api.APICore(subject,          complete=False,      configuration=None,
                                   db_file_name='api_call', is_parent=True)
```

Provides the logic and interface for the tests from the API.

Parameters

- **subject** (*str*) – The element we are testing.
- **complete** (*bool*) – Activate the return of a dictionary with significant - if not all - data about the test.
- **configuration** (*dict*) – The custom configuration to load.

Note: This will let you overwrite any of the previously given configuration index.

```
__APICore__inactive_database_management(subject, status)
```

Given the subject and status, we add or remove the subject from the inactive database.

```
availability(subject_type)
```

Make an availability check.

Parameters **subject_type** (*str*) – Should be one of the following.

- domain
- url

```
domain_and_ip()
```

Run a domain/IP availability check over the given subject.

```
domain_syntax()
```

Run a domain syntax check over the given subject.

```
ip_range_syntax()
```

Run an IP range syntax check over the given subject.

```
ip_syntax()
```

Run an IP syntax check over the given subject.

```
ipv4_range_syntax()
```

Run an IPv4 range syntax check over the given subject.

```
ipv4_syntax()
```

Run an IPv4 syntax check over the given subject.

```
ipv6_range_syntax()
```

Run an IPv6 range syntax check over the given subject.

```
ipv6_syntax()
```

Run an IPv6 syntax check over the given subject.

reputation (*subject_type*)

Make a reputation check.

Parameters **subject_type** (*str*) – Should be one of the following.

- domain
- url

subdomain_syntax ()

Run a subdomain syntax check over the given subject.

syntax (*subject_type*)

Make a syntax check.

Parameters **subject_type** (*str*) – Should be one of the following.

- domain
- url

url ()

Run an URL availability check over the given subject.

url_syntax ()

Run an IPv4 syntax check over the given subject.

CLICore ()

class `PyFunceble.core.cli.CLICore`

Provides some methods which are dedicated for the CLI.

classmethod `__CLICore__check_deprecated` (*upstream_version*)

Checks if the local version is deprecated.

classmethod `__CLICore__check_force_update` (*upstream_version*)

Checks if we need to force the update.

If it's the case, we will stop the tool.

classmethod `__CLICore__print_messages` (*upstream_version*)

Looks at the messages and prints the one that needs to be printed.

classmethod `colorify_logo` (*home=False*)

Print the colored logo based on global results.

Parameters **home** (*bool*) – Tell us if we have to print the initial coloration.

classmethod `compare_version_and_print_messages` ()

Compares the local with the upstream version.

classmethod `get_simple_coloration` (*status*)

Given a status we give the coloration for the simple mode.

Parameters **status** (*str*) – An official status output.

classmethod `get_upstream_version_file` ()

Provides the upstream version file.

classmethod `logs_sharing` ()

Prints an information message when the logs sharing is activated.

classmethod `print_header` ()

Prints the header if needed.

```
classmethod print_nothing_to_test ()  
    Prints the nothing to test message.  
  
classmethod save_into_database (output, filename)  
    Saves the current status inside the database.  
  
classmethod sort_generated_files ()  
    Sort the content of all files we generated.  
  
classmethod stay_safe ()  
    Prints a friendly message.
```

FileCore()

```
class PyFunceble.core.file.FileCore (file, file_content_type='domain')  
    Brain of PyFunceble for file testing.
```

Parameters

- **file** (*str*) – The file we are testing.
- **file_type** (*str*) – The file type. Should be one of the following.
 - domain
 - url

```
_FileCore__run_single_test (subject, ignore_inactive_db_check=False)  
    Run a test for a single subject.
```

```
_FileCore__test_line (line, ignore_inactive_db_check=False)  
    Tests a given line.
```

```
cleanup (auto_continue_db, auto_save, test_completed=False)  
    Runs the logic to run at the end of all test.
```

```
construct_and_get_shadow_file (file_stream, ignore_inactive_db_check=False)  
    Provides a path to a file which contain the list to file.
```

The idea is to do a comparison between what we already tested and what we still have to test.

```
classmethod download_link (input_file)  
    Downloads the file if it is an URL and return the name of the new file to test.
```

```
generate_complement_status_file (subject, status)  
    Generate the complements status files.
```

```
generate_files (include_entries_without_changes=False)  
    Generates all needed files.
```

```
generate_files_of_status (status, include_entries_without_changes=False)  
    Generates the status file of all subjects of the given status.
```

Parameters

- **status** (*str*) – A status to filter.
- **include_entries_without_changes** (*bool*) – Descriptive enough.

```
classmethod get_complements (auto_continue_db)  
    Generate a list of complements to test.
```

```
classmethod get_subjects (line)  
    Given a line, we give the list of subjects to test.
```

post_test_treatment (*test_output*, *file_content_type*, *complements_test_started=False*,
auto_continue_db=None, *inactive_db=None*, *mining=None*,
whois_db=None)

Do the post test treatment.

run_test ()

Run the test of the content of the given file.

classmethod should_be_ignored (*subject*, *auto_continue_db*, *inactive_db*, *ig-*
nore_inactive_db_check=False)

Given a subject which is supposed to be tested, we check if we should ignore it.

test (*subject*)

Tests the given subject and return it's results.

write_in_shadow_file_if_needed (*line*, *temp_file*, *ignore_inactive_db_check=False*, *autocon-*
tinue=None, *inactive_db=None*)

Checks if the given line should be in the shadow file.

OurProcessWrapper ()

class PyFunceble.core.multiprocess.**OurProcessWrapper** (*args, **kwargs)

Wrapper of Process. The object of this class is just to overwrite `Process.run()` in order to catch exceptions.

Note: This class takes the same arguments as `Process`.

exception

Provides a way to check if an exception was send.

run ()

Overwrites `Process.run()`.

MultiprocessCore ()

class PyFunceble.core.multiprocess.**MultiprocessCore** (*file*,
file_content_type='domain')

Brain of PyFunceble for file testing with multiple processes.s.

Parameters

- **file** (*str*) – The file we are testing.
- **file_type** (*str*) – The file type. Should be one of the following.
 - domain
 - url

__MultiprocessCore__check_exception (*processes*, *manager_data*)

Checks if an exception is present into the given pool of processes.

:param list processes. A list of running processes.

__MultiprocessCore__merge_processes_data (*manager_data*)

Reads all results and put them at the right location.

__MultiprocessCore__run_multiprocess_test (*stream*, *manager*, *ig-*
nore_inactive_db_check=False)

Tests the content of the given file.

`__MultiprocessCore__start_process` (*subject*, *manager_data*, *ignore_inactive_db_check=False*) *ig-*

Starts a new process.

`construct_and_get_shadow_file` (*file_stream*, *ignore_inactive_db_check=False*)

Provides a path to a file which contain the list to file.

The idea is to do a comparison between what we already tested and what we still have to test.

`run_test` ()

Runs the test of the content of the given file.

`test` (*subject*, *file_content_type*, *loader*, *manager_data*, *intern*, *ignore_inactive_db_check=False*, *custom=None*)

Tests the given subject and return the result.

`work_process` (*args)

Work process for `construct_and_get_shadow_file`.

`SimpleCore` ()

`class` `PyFunceble.core.simple.SimpleCore` (*subject*)

Brain of PyFunceble for simple test.

Parameters **`subject`** (*str*) – The subject we are testing.

`__SimpleCore__save_in_database` (*dataset*)

Saves the given dataset (result) into the database.

Parameters **`dataset`** (*dict*, *list*) – The dataset to save.

`domain` ()

Handles the simple domain testing.

`test` (*subject*, *subject_type*)

Processes a test of the given subject and return the result.

`url` ()

Handles the simple URL testing.

22.9 Database

22.9.1 Problematic

How can we store our most relevant information?

22.9.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.



Provides the database interfaces.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfuncible.github.io/special-thanks.html>

Contributors: <https://pyfuncible.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFuncible>

Project documentation: <https://pyfuncible.readthedocs.io/en/master/>

Project homepage: <https://pyfuncible.github.io/>

License:

```
Copyright 2017, 2018, 2019, 2020 Nissar Chababy

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

InactiveDB()

class PyFuncible.database.inactive.InactiveDB(filename, parent_process=False)

Provides the inactive database logic and interface.

Parameters filename (*str*) – The name of the file we are processing.

__InactiveDB__execute_query (query)

Executes the query to get the list to retest or already tested.

__merge ()

Merges the database with the older one which has already been set into the database.

add (subject, status)

Adds the given subject into the database.

Parameters

- **subject** (*str*) – The subject we are working with.
- **status** (*str*) – The status of the given subject.

classmethod authorization ()

Provides the execution authorization.

clean ()

Cleans everything which is not needed anymore.

classmethod datetime ()

Gets the timestamp where we are going to save our current list.

Returns The timestamp to append with the currently tested element.

Return type `intlstr`

get_already_tested()

Returns a set of already tested subjects.

classmethod get_table_name()

Returns the name of the table to use.

get_to_clean()

Returns a set of subject to clean from the database.

get_to_retest()

Returns a set of subject to retest.

initiate()

Initiates the databsse.

load()

Loads the content of the database file.

remove(subject)

Removes all occurrences of the given subject from the database.

Parameters `subject (str)` – The subject we are working with.

save()

Saves the current database into the database file.

WhoisDB()

class `PyFunceble.database.whois.WhoisDB(parent_process=False)`

Provides the WHOIS database interface and logic.

add(subject, expiration_date, record=None)

Adds the given subject and expiration date to the database.

Parameters

- **subject (str)** – The subject we are working with.
- **expiration_date (str)** – The extracted expiration date.
- **record (str)** – The WHOIS record.

classmethod authorization()

Provides the operation authorization.

get_expiration_date(subject)

Gets the expiration date of the given subject.

Parameters `subject (str)` – The subject we are working with.

Returns The expiration date from the database.

Return type `str|None`

classmethod get_table_name()

Returns the name of the table to use.

is_time_older(subject)

Checks if the expiration time of the given subject is older.

Parameters `subject (str)` – The subject we are working with.

Note: Of course, we imply that the subject is in the database.

load()

Loads the database file into the database.

classmethod merge(*old*)

Merges the older version of the database into the new version.

Parameters *old* (*dict*) – The old version of the database.

Returns The database in the new format.

Return type *dict*

save()

Saves the database into the database file.

22.10 Downloader

22.10.1 Problematic

How can we easily download one of our static files ?

22.10.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.

Provides the downloaders of all our static files.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfuncible.github.io/special-thanks.html>

Contributors: <https://pyfuncible.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFuncible>

Project documentation: <https://pyfuncible.readthedocs.io/en/master/>

Project homepage: <https://pyfuncible.github.io/>

License:

Copyright 2017, 2018, 2019, 2020 Nissar Chababy

Licensed under the Apache License, Version 2.0 (the "License");
you may **not** use this file **except in** compliance **with** the License.
You may obtain a copy of the License at

(continues on next page)

(continued from previous page)

```
http://www.apache.org/licenses/LICENSE-2.0
```

Unless required by applicable law **or** agreed to **in** writing, software distributed under the License **is** distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express **or** implied. See the License **for** the specific language governing permissions **and** limitations under the License.

DownloaderBase()

class PyFuncible.downloader.base.DownloaderBase

Provides the downloader base.

DOWNTIME_INDEX = None

Used to set/track the download time of the current file.

Return type str

REDOWNLOAD_AFTER = 1

Used to set the redownload frequency in day.

Return type int

get_all_downtimes()

Provides all download times.

Return type dict

get_downtime()

Provides the download timestamp of the current DOWNTIME_INDEX.

Return type dict, None

is_downtime_set()

Checks if the downtime is set for the current DOWNTIME_INDEX.

Return type bool

is_last_download_expired()

Checks if the last downloaded file is expired.

process()

Process the download and returns the downloaded text.

Return type str, None

save_downtimes()

Saves all downtimes.

update_downtime()

Updates the current download time.

ConfigDownloader()

class PyFuncible.downloader.config.ConfigDownloader

Provides the dowloader of the production configuration file.

DBTypeDownloader()

class PyFunceble.downloader.db_type.**DBTypeDownloader**
Provides the downloader of the desired database type file.

DirectoryStructureDownloader()

class PyFunceble.downloader.directory_structure.**DirectoryStructureDownloader**
Provides the downloader of the directory structure file.

IANADownloader()

class PyFunceble.downloader.iana.**IANADownloader**
Provides the downloader of the iana database file.

IPv4ReputationDownloader()

class PyFunceble.downloader.ipv4_reputation.**IPv4ReputationDownloader**
Provides the downloader of the IPv4 reputation file.

PublicSuffixDownloader()

class PyFunceble.downloader.publicsuffix.**PublicSuffixDownloader**
Provides the downloader of the public suffix list file.

22.11 Engine

22.11.1 Problematic

- How to auto continue?
- How to auto save?
- How to implement a proper debugger?
- How to get subjects related to a tested subject?
- How to work with different database types?
- How to sort our outputs?

22.11.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.



Provides the engine interfaces.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfunceble.github.io/special-thanks.html>

Contributors: <https://pyfunceble.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFunceble>

Project documentation: <https://pyfunceble.readthedocs.io/en/master/>

Project homepage: <https://pyfunceble.github.io/>

License:

```
Copyright 2017, 2018, 2019, 2020 Nissar Chababy
```

```
Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at
```

```
http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.
```

AutoContinue()

class PyFunceble.engine.auto_continue.AutoContinue(*filename, parent_process=False*)

Provides the auto-continue subsystem.

__AutoContinue__generate_complements()

Generates the complements from the given list of tested.

__AutoContinue__get_or_generate_complements_json()

Gets or generates the complements while working with as JSON formatted database.

__AutoContinue__get_or_generate_complements_mysql()

Gets or generates the complements while working with as MySQL/MariaDB formatted database.

add(*subject, status*)

Adds the given subject into the database.

classmethod authorization()

Provides the execution authorization.

clean()

Cleans the database.

get_already_tested()

Returns the list of subjects which were already tested as a set.

get_or_generate_complements()

Gets or generates the complements.

classmethod get_table_name()

Returns the name of the table to use.

is_empty()
Checks if the database related to the currently tested file is empty.

load()
Loads previously saved database.

save()
Saves the current state of the database.

update_counters()
Updates the counters.

AutoSave()

class PyFunceble.engine.auto_save.**AutoSave** (*start_time=None*)
Provides the autosave interface.

classmethod **get_current_ci()**
Provides the current CI to use.

is_time_exceed()
Checks if the end time is exceed.

process (*test_completed=False*)
Processes the autosave base on the current state of the test.

Parameters **test_completed** (*bool*) – Tell us if we finished the test.

Logger()

class PyFunceble.engine.logger.**Logger** (*debug=False, on_screen=False, output_directory=None*)
Provides our logging logic.

_Logger__get_handler (*handler_type*)
Provides a handler for of the given type.

_Logger__init_loggers()
Initiates all loggers.

_Logger__set_output_directory (*output_directory*)
Shares the given output directory.

Note: If the given output directory does not exists, we create it.

Parameters **output_directory** (*string*) – The output directory.

authorization (*debug*)
Provides the operation authorization.

critical (*message*)
Logs the critical message.

debug (*message*)
Logs the debug message.

error (*message*)
Logs the error message.

exception()

Logs the exception message.

fatal(*message*)

Logs the fatal message.

format_to_apply = '%(asctime)s::%(levelname)s::%(origin_path)s::%(origin_line)s@%(orig

The format to apply.

classmethod get_origin_info()

Returns the information about where the logger was triggered.

Returns

A tuple, which is composed of the following.

(trigger file path, trigger line, trigger function/method name)

Return type `tuple`

info(*message*)

Logs the info message.

init(*output_directory=None*)

Initiates the logger.

root_logger_format = '%(asctime)s::%(levelname)s] (PID%(process)s::%(processName)s) : %

The format to parse to the root logger (if used).

warning(*message*)

Logs the warning message.

Mining()

class `PyFunceble.engine.mining.Mining` (*filename*, *parent_process=False*)

Manages the minig subsystem.

classmethod authorization()

Provides the operation authorization.

classmethod get_history(*url*)

Gets the history of the given url.

Parameters *url* (*str*) – An URL to call.

Returns The list of links.

Return type `list`

classmethod get_table_name()

Returns the name of the table to use.

list_of_mined()

Provides the list of mined domains so that they can be tested.

Returns

The list of mined domains.

The returned format is the following:

```
[
    (index_to_delete_after_test, mined),
    (index_to_delete_after_test, mined),
    (index_to_delete_after_test, mined)
]
```

Return type `list`

load()

Loads the content of the database file.

mine (*subject*, *subject_type*)

Searches for domain or URL related to the original URL or domain. If some are found, we add them into the database.

Parameters

- **subject** (*str*) – The subject we are working with.
- **subject_type** (*str*) – The type of the subject.

Can me one of the following:

- `url`
- `domain`

remove (*subject*, *history_member*)

Removes the given subject from the database assigned to the currently tested file.

Parameters

- **subject** (*str*) – The subject we are working with.
- **history_member** (*str*) – The history member to delete.

save()

Saves the content of the database into the database file.

MySQL()

class `PyFunceble.engine.mysql.MySQL`

Provides our way to work with our mysql/mariadb database.

are_tables_present()

Checks if all our tables are present.

classmethod `authorization()`

Provides the authorization to operate.

create_tables_and_apply_patches()

Creates the tables of the database and apply the patches.

errors

alias of `pymysql.err.IntegrityError`

classmethod `get_int_cast_type()`

Provides the right integer casting.

init_post_connection()

Initiate everything needed after the connection.

init_pre_connection()

Initiate everything needed before the connection.

classmethod parse_env_file (*env_file_location*)

Parses the environment file into something we understand.

Parameters *env_file_location* (*str*) – The location of the file we have to parse.

classmethod parse_mysql_sql_file ()

Parses our mysql.sql file into something we understand.

save_to_env_file (*envs*, *env_file_location*)

Saves the given dict of environment variable into our environment file.

Parameters

- **envs** (*dict*) – A dict of environment variables to save.
- **env_file_location** (*str*) – The location of the file we have to update.

Sort()

class PyFunceble.engine.sort.Sort

Provides some sorting presets which we can parse to PyFunceble.helpers.list.List.custom_format.

classmethod hierarchical (*element*)

The idea behind this method is to sort a list of domain hierarchically.

Parameters *element* (*str*) – The element we are currently reading.

Returns The formatted element.

Return type *str*

Note: For a domain like `aaa.bbb.ccc.tdl`.

A normal sorting is done in the following order:

1. `aaa`
2. `bbb`
3. `ccc`
4. `tdl`

This method allow the sorting to be done in the following order:

1. `tdl`
2. `ccc`
3. `bbb`
4. `aaa`

classmethod standard (*element*)

Implements the standard and alphabetical sorting.

Parameters *element* (*str*) – The element we are currently reading.

Returns The formatted element.

Return type `str`

22.12 Extractor

22.12.1 Problematic

How can we extract data from other data?

22.12.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.

Provides the extractors.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfuncible.github.io/special-thanks.html>

Contributors: <https://pyfuncible.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFuncible>

Project documentation: <https://pyfuncible.readthedocs.io/en/master/>

Project homepage: <https://pyfuncible.github.io/>

License:

```
Copyright 2017, 2018, 2019, 2020 Nissar Chababy
```

```
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
```

```
http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

ExtractorBase()

class PyFuncible.extractor.base.**ExtractorBase**(data)

Provides the extractor base.

Parameters data (`str`) – The data to work with.

`get_extracted()`
Provides the converted data.

`ExpirationDate()`

class `PyFunceble.extractor.expiration_date.ExpirationDate(data)`
Provides a way to extract the expiration date from a given WHOIS record.

Parameters `data (str)` – The WHOIS record.s

`__ExpirationDate__extract_it()`
Try to extract the expiration date from the given data.

`__ExpirationDate__format_it(data)`
Formats the given data.

`__ExpirationDate__format_management(index, matched)`
A format management helper.

Note: Please note that the second value of the case represent the groups in order `[day, month, year]`. This means that a `[2, 1, 0]` will be for example for a date in format `2017-01-02` where `01` is the month.

Parameters

- **index** (`int`) – The identifiant of the regex.
- **matched** (`list`) – The matched result to format.

Returns A list representing the expiration date. The list can be “decoded” like `[day, month, year]`

Return type `list|None`

22.13 Helpers

22.13.1 Problematic

How can we avoid writing the same thing every time?

22.13.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.



Provides the helpers.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfunceble.github.io/special-thanks.html>

Contributors: <https://pyfunceble.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFunceble>

Project documentation: <https://pyfunceble.readthedocs.io/en/master/>

Project homepage: <https://pyfunceble.github.io/>

License:

```
Copyright 2017, 2018, 2019, 2020 Nissar Chababy

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

Command()

class PyFunceble.helpers.Command(*command*, *encoding*='utf-8')

Shell command execution.

Parameters

- **command** (*str*) – The command to execute.
- **encoding** (*str*) – The encoding to use to decode the shell output.

_decode_output (*to_decode*)

Decode the output of a shell command in order to be readable.

Parameters **to_decode** (*bytes*) – Output of a command to decode.

Returns The decoded output.

Return type *str*

execute ()

Execute the given command.

Returns The output of the command.

Return type *str*

run (*rstrip=True*)

Run the given command and yield each line(s) one by one.

Note: The difference between this method and *execute()* is that *execute()* wait for the process to end in order to return its output while this method return each line one by one - as they are outputed.

Parameters **rstrip** (*bool*) – Deactivates the rstrip of the output.

run_to_stdout()

Run the given command and print each line(s) to stdout.

Dict()

class PyFunceble.helpers.Dict(*main=None*)

Simplify some dict manipulation.

Parameters *main* (*dict*) – The main dict to work with.

Raises **TypeError** – When *main* is not a dict nor a list (tolerated).

classmethod **from_json** (*json_str*, *return_dict_on_error=True*)

Converts a given JSON string to dict/list.

Parameters **return_dict_on_error** (*bool*) – Return a dict instead of a NoneType.

Return type dictlist

classmethod **from_json_file** (*file_path*, *encoding='utf-8'*, *return_dict_on_error=True*)

Reads the given file path and convert it's content to dict/list (tolerated).

Parameters

- **file_path** (*str*) – The file path.
- **return_dict_on_error** (*bool*) – Return a dict instead of a NoneType.

Return type dictlist

classmethod **from_yaml** (*yaml_str*)

Converts the given YAML string to dict/list.

Parameters **yaml_str** (*str*) – The YAML string to convert.

Return type dictlist

classmethod **from_yaml_file** (*file_path*, *encoding='utf-8'*)

Converts a given YAML formatted file, into dict/list.

Parameters

- **file_path** (*str*) – The file path.
- **encoding** (*str*) – The encoding to use.

Return type dictlist

has_same_keys_as (*to_check*, *loop=False*)

A dirty solution which checks keys are presents in both given dict.

Parameters

- **to_check** (*dict*) – The dict to check.
- **loop** (*bool*) – DO NOT USE, only used to tell us wen to return the list of dataset or the final result.

Return type *bool*

remove_key (*key_to_remove*)

Remove a given key from a given dictionary.

Parameters **key_to_remove** (*list/str*) – The key(s) to delete.

Returns The dict without the given key(s).

Return type dict|None

rename_key (*key_to_rename*, *strict=True*)

Rename the given keys from the given dictionary.

Parameters

- **key_to_rename** (*dict*) – The key(s) to rename.
Expected format: {old:new}
- **strict** (*bool*) – Tell us if we have to rename the exact index or the index which looks like the given key(s)

Returns The well formatted dict.

Return type dict|None

to_json (*ensure_ascii=False*, *indent=4*, *sort_keys=True*)

Converts a given dict to JSON and return the json string.

Parameters

- **ensure_ascii** (*bool*) – Avoids unicode.
- **indent** (*int*) – The indentation to apply.
- **sort_keys** – Sort the keys.

Return type str

to_json_file (*file_path*, *ensure_ascii=False*, *indent=4*, *sort_keys=True*, *encoding='utf-8'*)

Converts the given dict to JSON and save the result into a given file path.

Parameters

- **file_path** (*str*) – The file path.
- **ensure_ascii** (*bool*) – Avoids unicode.
- **indent** (*int*) – The indentation to apply.
- **sortkeys** (*bool*) – Sorts the keys.

to_yaml (*encoding='utf-8'*, *default_flow_style=False*, *indent=4*, *allow_unicode=True*, *sort_keys=True*)

Converts the given dict/list to the YAML format and return the result.

Parameters

- **encoding** (*str*) – The encoding to use.
- **default_flow_style** (*bool*) – Uses the default flow style.
- **indent** (*int*) – The indentation to apply.
- **allow_unicode** (*bool*) – Allows the decoding of unicode chars.
- **sort_keys** (*bool*) – Sors the keys.

Return type dict|list

to_yaml_file (*file_path*, *encoding='utf-8'*, *default_flow_style=False*, *indent=4*, *allow_unicode=True*, *sort_keys=True*)

Converts the given dict/list to YAML and save the result into a file.

Parameters

- **file_path** (*str*) – The file path.

- **encoding** (*str*) – The encoding.
- **default_flow_style** (*bool*) – Uses the default flow style.
- **indent** (*int*) – The indentation to apply.
- **allow_unicode** (*bool*) – Allows the decoding of unicode chars.
- **sort_keys** (*bool*) – Sorts the keys.

Directory()

class PyFunceble.helpers.Directory(*dir_path=None*)

Simplify the directories manipulation.

:param str *dir_path* the path to work with.

create (*dir_path=None*)

Creates the given directory path.

Returns The output of `self.exists` after the directory creation.

Return type *bool*

delete (*dir_path=None*)

Deletes the given directory path.

Returns `not self.exists` after the directory deletion.

Rtypt *bool*

exists (*dir_path=None*)

Checks if the given directory exists.

fix_path (*dir_path=None, splited_path=None*)

Fix the path of the given path.

Note: We consider a path as fixed if it ends with the right directory separator.

classmethod **get_current** (*with_end_sep=False*)

Returns the current directory path.

Parameters **with_end_sep** (*bool*) – Add a directory separator at the end.

Download()

class PyFunceble.helpers.Download(*url, verify_certificate=True*)

Simplification of the downloads.

Parameters

- **url** (*str*) – The url to download.
- **verify_certificate** (*bool*) – Allows/Disallows the certificate verification.

text (*destination=None*)

Download the body of the given url.

Note: if *destination* is set to *None*, we only return the output.

Otherwise, we save the output into the given destination, but we also return the output.

Parameters `destination` (*str*) – The download destination.

Return type `str`:

Raises

- **Exception** – When the status code is not 200.
- **NoInternetConnection** – When no connection could be made.

File()

class `PyFunceble.helpers.File` (*file_path=None*)

Simplify the file manipulations.

Parameters `file_path` (*str*) – The file path to work with.

copy (*destination*)

Copy the globally given file path to the given destination.

Parameters `destination` (*str*) – The destination of the copy.

delete (*file_path=None*)

Deletes the given file path if it exists.

Parameters `file_path` (*str*) – The file path to check.

Note: If `None` is given, we report to the globally given file path.

Returns The non existence state of the file.

Return type `bool`

exists (*file_path=None*)

Checks if the given file path exists.

Parameters `file_path` (*str*) – The file path to check.

Note: If `None` is given, we report to the globally given file path.

Return type `bool`

get_size (*file_path=None*)

Provides the size (in bytes) of the given file path.

Parameters `file_path` (*str*) – The file path to check.

Note: If `None` is given, we report to the globally given file path.

Return type `int`

is_empty (*file_path=None*)

Checks if the given file path is empty.

move (*destination*)

Move the globally given file path to the given destination.

Parameters **destination** (*str*) – The destination of the file.

read (*file_path=None, encoding='utf-8'*)

Read the given file path and return it's content.

Parameters

- **file_path** (*str*) – The file path to check.

Note: If None is given, we report to the globally given file path.

- **encoding** (*str*) – The encoding to use.

Return type *str*

write (*data, overwrite=False, encoding='utf-8', file_path=None*)

Write the given data into the given file path.

Parameters

- **data** (*str*) – The data to write.
- **encoding** (*str*) – The encoding to use while opening the file.
- **file_path** (*str*) – The file path to check.

Note: If None is given, we report to the globally given file path.

Hash()

class PyFunceble.helpers.**Hash** (*algo='sha512_224'*)

Simplify the hashing of data or file content.

Parameters **algo** (*str*) – The algorithm to use for hashing.

Raises **ValueError** – When the given algo is not known.

data (*data*)

Hash the given data.

Parameters **data** (*str, bytes*) – The data to hash.

Return type *str*

file (*file_path, encoding='utf-8'*)

Open the given file, and it's content.

Parameters **file_path** (*str*) – The file to hash.

Return type *str*

List()

class PyFunceble.helpers.**List** (*main, remove_empty=False*)

Simplify the list manipulation.

Parameters

- **main** (*list*) – The main list to work with.
- **remove_empty** (*bool*) – Process the deletion of empty strings.

custom_format (*key_method*, *reverse=False*)

Return a well formatted list. With the *key_method* as a function/method to format the elements before sorting.

Parameters

- **key_method** (*function/method*) – A function or method to use to format the readed element before sorting.
- **reverse** (*bool*) – Tell us if we have to reverse the list.

Returns A sorted list.

Return type *list*

format ()

Return a well formatted list. Basicaly, it's sort a list and remove duplicate.

Returns A sorted, without duplicate, list.

Return type *list*

Merge()

class PyFunceble.helpers.**Merge** (*main*)

Simplify the merging of dict and list.

Parameters **main** (*str*, *tuple*, *bool*, *int*, *dict*, *list*, *float*) – The main data to work with.

_Merge__dict (*origin*, *strict=True*)

Process the dict merging.

Parameters **strict** (*bool*) – Activates the strict mode.

Return type *dict*

_Merge__list (*origin*, *strict=True*)

Process the list merging.

Parameters **strict** (*bool*) – Activates the strict mode.

Return type *list*

into (*origin*, *strict=True*)

Process the mergin.

Parameters

- **origin** – The original data.
- **strict** (*bool*) – Activates the strict mode.

Regex()

class PyFunceble.helpers.**Regex** (*regex*, *escape=False*)

Simplify the regex matching and usage.

Parameters

- **regex** (*str*) – The regex to use.
- **escape** (*bool*) – Escapes the given regex.

get_matching_list (*data*)

Returns the strings which does the match the regex in the given data.

get_not_matching_list (*data*)

Returns the strings which does not the match the regex in the given data.

match (*data*, *rematch=False*, *group=0*, *return_match=True*)

Checks if the given data match the given regex string.

Parameters

- **data** (*str*) – The data to work with.
- **rematch** (*bool*) – The equivalent of the \$BASH_REMATCH but in Python.
It's basically a list of all groups.
- **group** (*bool*) – The group to return when *return_match* is set to *True*.
- **return_match** (*bool*) – Return the part that match the given regex string.

replace_match (*data*, *replacement*, *occurences=0*, *multiline=False*)

Replaces the string which match the regex string with the given replacement.

Parameters

- **data** (*str*) – The data to work with.
- **replacement** (*str*) – The replacement of the matched regex.
- **occurences** (*int*) – The number of occurences to replace.

Note: 0 means all occurences.

Return type *str*

22.14 Lookup

22.14.1 Problematic

How can we look for information about a given subject ?

22.14.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.



Provides the lookup interfaces.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfuncible.github.io/special-thanks.html>

Contributors: <https://pyfuncible.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFuncible>

Project documentation: <https://pyfuncible.readthedocs.io/en/master/>

Project homepage: <https://pyfuncible.github.io/>

License:

```
Copyright 2017, 2018, 2019, 2020 Nissar Chababy
```

```
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
```

```
http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

DNSLookup()

class PyFuncible.lookup.dns.DNSLookup(*dns_server=None, lifetime=3, tcp=False*)
DNS lookup interface.

Parameters

- **subject** (*str*) – The subject we are working with.
- **dns_server** (*list/tuple/str*) – The DNS server we are working with.
- **lifetime** (*int*) – Set the lifetime of a query.

__DNSLookup__get_dns_servers_from (*inputed_dns*)

Given a list or an input representing dns servers, we ensure that we have a list of a string which represent an IP.

Parameters **input** (*str, list, tuple*) – The inputed DNS server(s).

Returns The list of dns server to use.

Return type *list*

classmethod **__DNSLookup__get_resolver** (*dns_server*)

Provides the configured dns resolver.

__DNSLookup__get_server_and_port_from (*inputed_dns*)

Given a list or an input representing dns server, we split the server from the port.

Parameters **inputed_dns** – The inputed DNS server(s).

Returns

A tuple with:

(The DNS, the port)

Return type `tuple`

`_DNSLookup__request_complete_not_ip` (*subject*, *tcp=None*)
Requests and provides the complete DNS spectrum.

Return type `dict`

`_DNSLookup__request_ip` (*subject*, *tcp=None*)
Handle the request for a subject which is an IP.

Parameters

- **subject** (*str*) – The subject we are working with.
- **tcp** (*bool*) – Tell us to use TCP for query.

Return type `dict`

Raises **ValueError** – When a non string *subject* is given.

`_DNSLookup__request_not_ip` (*subject*, *complete=False*, *tcp=None*)
Handle the request for a subject which is not an IP.

Return type `dict`

Raises **ValueError** – When a non string *subject* is given.

`a_record` (*subject*, *tcp=None*)
Return the A record of the given subject (if found).

Parameters

- **subject** (*str*) – The subject we are working with.
- **tcp** (*bool*) – Tell us to use TCP for query.

Returns A list of A record(s).

Return type `list`, `None`

Raises **ValueError** – When a non string *subject* is given.

`aaaa_record` (*subject*, *tcp=None*)
Return the AAAA record of the given subject (if found).

Parameters

- **subject** (*str*) – The subject we are working with.
- **tcp** (*bool*) – Tell us to use TCP for query.

Returns A list of A record(s).

Return type `list`, `None`

Raises **ValueError** – When a non string *subject* is given.

`cname_record` (*subject*, *tcp=None*)
Return the CNAME record of the given subject (if found).

Parameters

- **subject** (*str*) – The subject we are working with.
- **tcp** (*bool*) – Tell us to use TCP for query.

Returns A list of CNAME record(s).

Return type `list`, `None`

Raises **ValueError** – When a non string `subject` is given.

dname_record (*subject*, *tcp=None*)

Return the DNAME record of the given subject (if found).

Parameters

- **subject** (*str*) – The subject we are working with.
- **tcp** (*bool*) – Tell us to use TCP for query.

Returns A list of DNAME record(s).

Return type `list`, `None`

Raises **ValueError** – When a non string `subject` is given.

classmethod get_addr_info (*subject*)

Get and return the information of the given subject (address).

Parameters **subject** (*str*) – The subject we are working with.

Returns A list of address.

Return type `list`, `None`

Raises **ValueError** – When a non string `subject` is given.

classmethod get_host_by_addr (*subject*)

Get and return the host of the given subject (address).

Parameters **subject** (*str*) – The subject we are working with.

Returns

A dict in the following format or None.

```
{
    "hostname": "",
    "aliases": [],
    "ips": []
}
```

Return type `dict`, `None`

Raises **ValueError** – When a non string `subject` is given.

classmethod is_record_present_in_result (*to_check*, *result*, *allow_empty=False*)

Checks if the given record type is in the result.

Parameters

- **to_check** (*list*, *str*, *tuple*) – The record to check the presence.
- **result** (*dict*) – The result to work with.
- **allow_empty** (*bool*) – Tell to if we allow and empty result as present.

Return type `bool`

mx_record (*subject*, *tcp=None*)

Return the MX record of the given subject (if found).

Parameters **subject** (*str*) – The subject we are working with.

Returns A list of MX record(s).

Return type `list`, `None`

Raises **ValueError** – When a non string `subject` is given.

ns_record (*subject*, *tcp=None*)

Return the NS record of the given subject (if found).

Parameters

- **subject** (*str*) – The subject we are working with.
- **tcp** (*bool*) – Tell us to use TCP for query.

Returns A list of NS record(s).

Return type `list`, `None`

Raises **ValueError** – When a non string `subject` is given.

ptr_record (*subject*, *reverse_name=True*, *tcp=None*)

Return the PTR record of the given subject (if found).

Parameters

- **subject** (*str*) – The subject we are working with.
- **tcp** (*bool*) – Tell us to use TCP for query.

Returns A list of PTR record(s).

Return type `list`, `None`

Raises **ValueError** – When a non string `subject` is given.

request (*subject*, *complete=False*, *tcp=None*)

Perform a DNS lookup/requests.

Parameters

- **subject** (*str*) – The subject we are working with.
- **complete** (*bool*) – Tell us to return as many result as possible.
- **tcp** (*bool*) – Tell us to use TCP for query.

Returns

A dict with following index if the given subject is not registered into the given DNS server.
(More likely local subjects).

```
{
    "hostname": "",
    "aliases": [],
    "ips": []
}
```

A dict with following index for everything else (and if found).

```
{
    "nameservers": [],
    "A": [],
    "AAAA": [],
    "CNAME": [],
    "DNAME": [],
    "MX": [],
    "NS": [],
    "TXT": [],

```

(continues on next page)

(continued from previous page)

```

    "PTR": []
}

```

Return type dict**Raises** **ValueError** – When a non string `subject` is given.**txt_record** (*subject*, *tcp=None*)

Return the TXT record of the given subject (if found).

Parameters

- **subject** (*str*) – The subject we are working with.
- **tcp** (*bool*) – Tell us to use TCP for query.

Returns A list of TXT record(s).**Return type** list, None**Raises** **ValueError** – When a non string `subject` is given.**update_lifetime** (*lifetime*)

Updates the lifetime of a query.

update_nameserver (*nameserver*)

Updates the nameserver to query.

HTTPCode()**class** PyFunceble.lookup.http_code.**HTTPCode** (*subject*, *subject_type*)

Get and return the HTTP code status of a given domain.

Parameters

- **subject** (*str*) – The subject we are working with.
- **_type** (*subject*) – The type of the subject we are working with. Should be one of the following.
 - url
 - domain
 - file_url
 - file_domain
 - ipv6

_get_it()

Get the HTTP code status.

Returns The matched HTTP status code.**Return type** int!None**get()**

Return the HTTP code status.

Returns The matched and formatted status code.**Return type** str!int!None

Iana()

```
class PyFunceble.lookup.iana.Iana
```

Let us interact with the IANA database.

```
    _Iana__get_content()
```

Provides the content of our IANA database.

PublicSuffix()

```
class PyFunceble.lookup.publicsuffix.PublicSuffix
```

Let us interact with the public suffix database.

```
    _PublicSuffix__get_content()
```

Provides the content of our PSL database.

Referer()

```
class PyFunceble.lookup.referer.Referer(subject)
```

Get the WHOIS server (referer) of the current domain extension according to the IANA database.

Parameters **subject** (*str*) – The subject we are working with.

```
    get()
```

Return the referer aka the WHOIS server of the current domain extension.

Returns

- [0] None if there is no referer.
- [0] **False** if the extension is unknown which implicitly means that the subject is INVALID
- [0] *str* The resolved IP to use.
- [1] *str*, None the domain referer.

Return type *tuple*

HostSSLAdapter()

```
class PyFunceble.lookup.requests.HostSSLAdapter(pool_connections=10,  
                                                  pool_maxsize=10,    max_retries=0,  
                                                  pool_block=False)
```

Extends the build-in HTTP Adapter for urllib3 for our needs.

```
    _HostSSLAdapter__resolve_with_cache(hostname)
```

Resolve the IP of the given hostname.

Parameters **hostname** (*str*) – The hostname to resolve.

Returns The IP of the host name or the hostname itself.

Return type *None, str*

```
    classmethod resolve(hostname)
```

Resolve the IP of the given hostname.

Parameters **hostname** (*str*) – The hostname to resolve.

Returns The IP of the host name or the hostname itself.

Return type `None, str`

send (*request*, ***kwargs*)

Overwrite the upstream :code:send' method.

We basically do the same. We only ensure that we request the IP from the chosen DNS record.

Parameters

- **request** – The PreparedRequest being sent.
- **stream** – (optional) Whether to stream the request content.
- **timeout** (*float or tuple or urllib3 Timeout object*) – (optional) How long to wait for the server to send data before giving up, as a float, or a (`connect timeout`, `read timeout`) tuple.
- **verify** – (optional) Either a boolean, in which case it controls whether we verify the server's TLS certificate, or a string, in which case it must be a path to a CA bundle to use
- **cert** – (optional) Any user-provided SSL certificate to be trusted.
- **proxies** – (optional) The proxies dictionary to apply to the request.

Return type `requests.Response`

HostAdapter()

class `PyFunceble.lookup.requests.HostAdapter` (*pool_connections=10, pool_maxsize=10, max_retries=0, pool_block=False*)

Extends the build-in HTTP Adapter for urllib3 for our needs.

`__HostAdapter__ resolve_with_cache` (*hostname*)

Resolve the IP of the given hostname.

Parameters **hostname** (*str*) – The hostname to resolve.

Returns The IP of the host name or the hostname itself.

Return type `None, str`

send (*request*, ***kwargs*)

Overwrite the upstream :code:send' method.

We basically do the same. We only ensure that we request the IP from the chosen DNS record.

Parameters

- **request** – The PreparedRequest being sent.
- **stream** – (optional) Whether to stream the request content.
- **timeout** (*float or tuple or urllib3 Timeout object*) – (optional) How long to wait for the server to send data before giving up, as a float, or a (`connect timeout`, `read timeout`) tuple.
- **verify** – (optional) Either a boolean, in which case it controls whether we verify the server's TLS certificate, or a string, in which case it must be a path to a CA bundle to use.
- **cert** – (optional) Any user-provided SSL certificate to be trusted.
- **proxies** – (optional) The proxies dictionary to apply to the request.

Return type `requests.Response`

IPv4Reputation

class PyFunceble.lookup.ipv4_reputation.IPv4Reputation

Let us checks a given IPv4 against the IPv4 reputation file.

`__IPv4Reputation__is_present` (*subject*)

Checks the given subject against the input file.

Requests()

class PyFunceble.lookup.requests.Requests

Handles all usage of requests.

Parameters **url** (*str*) – The URL to work with.

delete (*url*, ***kwargs*)

Sends a DELETE request. Returns Response object.

Parameters

- **url** – URL for the new Request object.
- **kwargs** – Optional arguments that `request` takes.

Return type `requests.Response`

exceptions = <module 'requests.exceptions' from '/home/docs/checkouts/readthedocs.org/

get (*url*, ***kwargs*)

Sends a GET request. Returns Response object.

Parameters

- **url** – URL for the new Request object.
- **kwargs** – Optional arguments that `request` takes.

Return type `requests.Response`

head (*url*, ***kwargs*)

Sends a HEAD request. Returns Response object.

Parameters

- **url** – URL for the new Request object.
- **kwargs** – Optional arguments that `request` takes.

Return type `requests.Response`

options (*url*, ***kwargs*)

Sends a OPTIONS request. Returns Response object.

Parameters

- **url** – URL for the new Request object.
- **kwargs** – Optional arguments that `request` takes.

Return type `requests.Response`

patch (*url*, ***kwargs*)

Sends a PATCH request. Returns Response object.

Parameters

- **url** – URL for the new `Request` object.
- **kwargs** – Optional arguments that `request` takes.

Return type `requests.Response`

post (*url*, ***kwargs*)

Sends a POST request. Returns `Response` object.

Parameters

- **url** – URL for the new `Request` object.
- **kwargs** – Optional arguments that `request` takes.

Return type `requests.Response`

put (*url*, ***kwargs*)

Sends a PUT request. Returns `Response` object.

Parameters

- **url** – URL for the new `Request` object.
- **kwargs** – Optional arguments that `request` takes.

Return type `requests.Response`

WhoisLookup()

class `PyFunceble.lookup.whois.WhoisLookup` (*subject*, *server=None*, *timeout=3*)

Implementation of the UNIX *whois* command.

Parameters

- **subject** (*str*) – The subject we are working with.
- **server** (*str*) – The WHOIS server we are working with.

Note: If `None` is given, we look for the best one.

- **timeout** (*int*) – The timeout to apply.

Warning: The timeout must be a modulo of 3.

request ()

Perform the WHOIS request.

22.15 Output

22.15.1 Problematic

How can we generate or work with outputs ?

22.15.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.

Provides the output related interfaces.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfunceble.github.io/special-thanks.html>

Contributors: <https://pyfunceble.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFunceble>

Project documentation: <https://pyfunceble.readthedocs.io/en/master/>

Project homepage: <https://pyfunceble.github.io/>

License:

```
Copyright 2017, 2018, 2019, 2020 Nissar Chababy
```

```
Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at
```

```
http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.
```

Clean()

```
class PyFunceble.output.clean.Clean(clean_all=False,file_path=None)
```

Provide the cleaning logic.

Note: By cleaning we mean the cleaning of the output directory.

Parameters

- **list_to_test** (*list* | *None*) – The list of domains we are testing.
- **clean_all** (*bool*) – Tell the subsystem if we need to clean all. Which include, of course, the output directory but also all other file(s) generated by our system.
- **file_path** (*str*) – The path to the file we tested.

Note: This is only relevant if you use the MariaDB/MySQL database.

almost_everything (*clean_all=False, file_path=False*)

Delete almost all discovered files.

Parameters **clean_all** (*bool*) – Tell the subsystem if we have to clean everything instead of almost everything.

classmethod **databases_to_delete** ()

Set the databases files to delete.

classmethod **file_to_delete** (*all_files=False*)

Return the list of file to delete.

Constructor ()

class `PyFunceble.output.constructor.Constructor` (*production=False*)

Basically a backup/reconstructor of our output directory.

Parameters **production** (*bool*) – Tell the subsystem if we are preparing for production which imply the execution of the backup instead of the “reconstructore” mode.

classmethod **_create_directory** (*directory, loop=False*)

Creates the given directory if it does not exist.

Parameters

- **directory** (*str*) – The directory to create.
- **loop** (*bool*) – Tell us if we are in the creation loop or not.

_get_structure ()

Get the structure we are going to work with.

Returns The structure we have to work with.

Return type `dict`

_restore_replace ()

Check if we need to replace “.gitignore” to “.keep”.

Returns The replacement status.

Return type `bool`

_update_structure_from_config (*structure*)

Update the paths according to configs.

Parameters **structure** (*dict*) – The read structure.

backup ()

Backup the developer state of *output/* in order to make it restorable and portable for user.

delete_unneeded ()

Delete the directory which are not registered into our structure.

restore ()

Restore the ‘output/’ directory structure based on the *dir_structure.json* file.

Generate()

```
class PyFunceble.output.generate.Generate(subject, subject_type, status, source=None, expiration_date=None, http_status_code=None, whois_server='Unknown', filename=None, ip_validation=False, end=False)
```

Generate different sort of files.

Parameters

- **subject** (*str*) – The subject we are working with.
- **subject_type** (*str*) – The type of the subject.
- **status** (*str*) – The caught status.
- **source** (*str*) – The source of the given status.
- **expiration_date** (*str*) – The expiration date of the domain (if caught).
- **http_status_code** (*str/int*) – The HTTP status code.
- **whois_server** (*str*) – The whois server.
- **filename** (*str*) – The name of the file we are testing.
- **ip_validation** (*bool*) – The IP validation check of the currently written subject.

__Generate__get_info_files_destinations (*output_hosts, output_domains, output_json*)

Given the output directory, this method return several paths.

Note: The given output directories have to be partially completed.

Indeed, we only do `output % final_location`.

Returns

The following paths:

```
(
    hosts_destination,
    plain_destination,
    json_destination,
    splited_destination
)
```

Return type *tuple*

__Generate__info_files_authorization()

Provide the authorization for the generation of info files.

Basicaly here is what we check:

- We are not testing as an imported module.

and

- The hosts file generation is activated.

or

- The plain list generation is activated.

or

- The “api_file_generation” was set into the CONFIGURATION.

`_analytic_host_file_directory()`

Return the analytic directory to write depending of the matched status.

`_do_not_produce_file()`

Check if we are allowed to produce a file based from the given information.

Returns The state of the production. True: We do not produce file. False: We do produce file.

Return type `bool`

`_prints_status_screen()`

Logic behind the printing (on screen) when generating status file.

`analytic_file(new_status, old_status=None)`

Generate Analytic/* files based on the given old and new statuses.

Parameters

- **`new_status`** (`str`) – The new status of the domain.
- **`old_status`** (`str`) – The old status of the domain.

`complements_file()`

Generate complements files base on the current status.

`info_files()`

Generate the hosts file, the plain list, the JSON file and the splitted files.

`prints_status_file()`

Logic behind the printing (in file) when generating status file.

`status_file(exclude_file_generation=False)`

Generate a file according to the domain status.

Parameters **`exclude_file_generation`** (`bool`) – A shorthand to disable any file generation.

`unified_file()`

Generate unified file. Understand by that that we use an unified table instead of a separate table for each status which could result into a misunderstanding.

`Logs()`

`class PyFunceble.output.logs.Logs(output=None)`

Provide a clean and unique way to work with logs. Indeed, it’s not good to have logs spread around the code :smile:

Parameters **`output`** (`str`) – A path to the JSON file we are going to write.

`classmethod _get_content(file)`

Get and return the content of the given log file.

Parameters **`file`** (`str`) – The file we have to get the content from.

:return The content of the given file. :rtype: dict

`classmethod _write_content(content,file)`

Write the content into the given file.

Parameters

- **content** (*str*) – The dict to write.
- **file** (*str*) – The file to write.

expiration_date (*subject, extracted*)

Logs the extracted expiration date.

Parameters

- **subject** (*str*) – The currently tested subject.
- **extracted** (*str*) – The extracted expiration date (from WHOIS record).

referrer_not_found (*subject, extension*)

Logs the case that the referer was not found.

Parameters

- **subject** (*str*) – The currently tested subject.
- **extension** (*str*) – The extension of the domain we are testing.

whois (*subject, record*)

Logs the WHOIS record if needed.

Parameters

- **subject** (*str*) – The currently tested subject.
- **record** (*str*) – The record to log.

Percentage()

class PyFunceble.output.percentage.**Percentage** (*domain_status=None, init=None*)

Calculation of the percentage of each status.

Parameters

- **domain_status** (*str*) – The status to increment.
- **init** (*dict*) – The data from a previous session we are continuing.

Warning: We expect the numbers and not the percentages.

classmethod **calculate** ()

Calculate the percentage of each status.

count ()

Count the number of domain for each status.

log ()

Print on screen and on file the percentages for each status.

Prints()

class PyFunceble.output.prints.**Prints** (*to_print, template, output_file=None, only_on_file=False*)

Print data on screen and into a file if needed. Template Possibilities: Percentage, Less, HTTP and any status you want.

Parameters

- **to_print** (*list*) – The list of data to print.
- **template** (*str*) – The template to use.

Note: Available templates:

- Percentage
 - Less
 - HTTP
 - any of the official status.
-

- **output_file** (*str*) – The path to the file to write.
- **only_on_file** (*bool*) – Tell us if we only have to print on file and not on screen.

__Prints__get_print_size()
Provides the size of the element to print.

__colorify (*data*)
Return colored string.

Parameters **data** (*str*) – The string to colorify.

Returns A colored string.

Return type *str*

__json_print ()
Management of the json template.

classmethod __size_from_header (*header*)
Get the size of each columns from the header.

Parameters **header** (*dict*) – The header template we have to get the size from.

Returns The maximal size of the each data to print.

Return type *list*

before_header ()
Print informations about PyFunceble and the date of generation of a file into a given path, if doesn't exist.

data ()
Management and input of data to the table.

Raises

Exception When self.data_to_print is not a list.

data_constructor (*size*)
Construct the table of data according to given size.

Parameters **size** (*list*) – The maximal length of each string in the table.

Returns A dict with all information about the data and how to which what maximal size to print it.

Return type OrderedDict

Raises

Exception If the data and the size does not have the same length.

header (*do_not_print=False*)

Management and creation of templates of header. Please consider as “header” the title of each columns.

Parameters **do_not_print** (*bool*) – Tell us if we have to print the header or not.

classmethod header_constructor (*data_to_print, header_separator='-', column_separator=''*)

Construct header of the table according to template.

Parameters

- **data_to_print** (*list*) – The list of data to print into the header of the table.
- **header_separator** (*str*) – The separator to use between the table header and our data.
- **colomnn_separator** (*str*) – The separator to use between each colomns.

Returns The data to print in a list format.

Return type *list*

22.16 Status

22.16.1 Problematic

How can we gather, manage and provide a status for a given subject ?

22.16.2 Documentation

The tool to check the availability or syntax of domain, IP or URL.



Provides the status interfaces.

Author: Nissar Chababy, @funilrys, contactTATAfunilrysTODTODcom

Special thanks: <https://pyfuncible.github.io/special-thanks.html>

Contributors: <https://pyfuncible.github.io/contributors.html>

Project link: <https://github.com/funilrys/PyFuncible>

Project documentation: <https://pyfuncible.readthedocs.io/en/master/>

Project homepage: <https://pyfuncible.github.io/>

License:

Copyright 2017, 2018, 2019, 2020 Nissar Chababy

Licensed under the Apache License, Version 2.0 (the "License");
you may **not** use this file **except in** compliance **with** the License.

(continues on next page)

(continued from previous page)

You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law **or** agreed to **in** writing, software distributed under the License **is** distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express **or** implied. See the License **for** the specific language governing permissions **and** limitations under the License.

GathererBase()

```
class PyFunceble.status.gatherer_base.GathererBase(subject, filename=None,
                                                    whois_db=None, inactive_db=None)
```

Provides the gatherer base.

Parameters

- **subject** (*str*) – The subject to gather data for.
- **filename** (*str*) – The filename which the subject is associated with.
- **whois_db** – An instance of the whois db interface.
- **inactive_db** – An instance of the inactive db interface.

```
gather_http_status_code()
```

Univertially gather the status code.

```
get()
```

Provides the status.

Status()

```
class PyFunceble.status.status.Status(subject)
```

The status object.

Provides the result of one of the status gatherer.

Parameters **subject** (*str*) – The subject which we describe.

Variables

- **_status_source** (*str*) – The status source before the application of our extra rules (if activated).
- **_status** (*str*) – The status before the application of our extra rules (if activated).
- **dns_lookup** (*list*) – The result of the DNS Lookup logic.
- **domain_syntax_validation** (*bool*) – The domain syntax validation status.
- **expiration_date** (*str*) – The expiration date of the subject.
- **http_status_code** (*str*) – The HTTP status code of the subject.
- **ipv4_range_syntax_validation** (*bool*) – The IPv4 range validation status.
- **ipv4_syntax_validation** (*bool*) – The IPv4 syntax validation status.

- **ipv6_range_syntax_validation** (*bool*) – The IPv6 range validation status.
- **ipv6_syntax_validation** (*bool*) – The IPv6 syntax validation status.
- **status_source** (*str*) – The final status source.
- **status** (*str*) – The final status.
- **subdomain_syntax_validation** (*bool*) – The subdomain syntax validation status.
- **tested** (*str*) – The tested subject.
- **url_syntax_validation** (*bool*) – The URL syntax validation status.
- **whois_server** (*str*) – The WHOIS server we contact-ed to get the WHOIS record.
- **whois_record** (*str*) – The WHOIS record.

get ()
Provides the status in a dict format.

Availability: **DomainAndIp** ()

```
class PyFunceble.status.availability.domain_and_ip.DomainAndIp(subject,      file-
                                                                name=None,
                                                                whois_db=None,
                                                                inac-
                                                                tive_db=None)
```

Gather the availability of the given IP or Domain.

__DomainAndIp__gather ()
Process the gathering.

__DomainAndIp__gather_expiration_date ()
Gather the expiration date.

__DomainAndIp__gather_extra_rules ()
Handle the lack of WHOIS record.

Availability: **ExtraRules** ()

```
class PyFunceble.status.availability.extra_rules.ExtraRules(subject,      sub-
                                                                ject_type,
                                                                http_status_code)
```

Manage some extra rules.,

Parameters

- **subject** (*str*) – The subject we are working with.
- **subject_type** (*str*) – The type of the subject we are working with. Should be one of the following.
 - domain
 - url
- **http_status_code** (*str/int*) – The extracted status code.

__ExtraRules__blogspot ()
Handle the blogspot SPECIAL case.

Returns (new status, new source) or None if there is any change to apply.

Return type tuple|None

`__ExtraRules__handle_http_code(previous_state)`

Handle the HTTP Code status escalation.

Parameters `previous_state` (*str*) – The previously caught status.

Returns (new status, new source) or None if there is any change to apply.

Return type tuple|None

`__ExtraRules__handle_ip_range()`

Handle the IP range status escalation.

Returns (new status, new source) or None if there is any change to apply.

Return type tuple|None

`__ExtraRules__handle_potentially_inactive(previous_state)`

Handle the potentially inactive case.

Parameters `previous_state` (*str*) – The previously caught status.

Returns (new status, new source) or None if there is any change to apply.

Return type tuple|None

`__ExtraRules__handle_potentially_up()`

Handle the potentially up case.

Returns (new status, new source) or None if there is any change to apply.

Return type tuple|None

`__ExtraRules__handle_reputation()`

Handle the reputation escalation.

Returns (new status, new source) or None if there is any change to apply.

Return type tuple|None

`classmethod __ExtraRules__http_status_code_up()`

Set what we return for the HTTP Code status escalation.

Returns (new status, new source)

Return type tuple

`classmethod __ExtraRules__special_down()`

Set what we return for the SPECIAL status de-escalation.

Returns (new status, new source)

Return type tuple

`__ExtraRules__special_down_404()`

Set what we return for the SPECIAL status de-escalation when the 404 status code is caught.

Returns (new status, new source)

Return type tuple

`classmethod __ExtraRules__special_up()`

Set what we return for the SPECIAL status escalation.

Returns (new status, new source)

Return type tuple

`_ExtraRules__wordpress_dot_com()`

Handle the wordpress.com SPECIAL case.

Returns (new status, new source) or None if there is any change to apply.

Return type tuple|None

handle (previous_state, previous_source)

Globally handle the case of the currently tested domain.

Availability: `Url()`

class PyFunceble.status.availability.url.**Url** (subject, filename=None, whois_db=None, inactive_db=None)

Gather the availability of the given URL.

`_Url__gather()`

Process the gathering.

Reputation: `DomainAndIp()`

class PyFunceble.status.reputation.domain_and_ip.**DomainAndIp** (subject, filename=None, whois_db=None, inactive_db=None)

Gather the reputation of the given domain or IP.

`_DomainAndIp__gather()`

Process the gathering.

Reputation: `Url()`

class PyFunceble.status.reputation.url.**Url** (subject, filename=None, whois_db=None, inactive_db=None)

Gather the reputation of the given URL (base).

`_Url__gather()`

Process the gathering.

Syntax: `DomainAndIp()`

class PyFunceble.status.syntax.domain_and_ip.**DomainAndIp** (subject, filename=None, whois_db=None, inactive_db=None)

Gather the availability of the given IP or Domain.

`_DomainAndIp__gather()`

Process the gathering.

Syntax: `Url()`

class PyFunceble.status.syntax.url.**Url** (subject, filename=None, whois_db=None, inactive_db=None)

Gather the syntax of the given URL.

`_Url_gather()`
Process the gathering.

CHAPTER 23

Indices and tables

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

p

- `PyFunceble`, [109](#)
- `PyFunceble.abstracts`, [153](#)
- `PyFunceble.check`, [149](#)
- `PyFunceble.cli`, [157](#)
- `PyFunceble.config`, [160](#)
- `PyFunceble.converter`, [162](#)
- `PyFunceble.core`, [165](#)
- `PyFunceble.database`, [170](#)
- `PyFunceble.downloader`, [173](#)
- `PyFunceble.engine`, [175](#)
- `PyFunceble.exceptions`, [151](#)
- `PyFunceble.extractor`, [181](#)
- `PyFunceble.helpers`, [182](#)

Symbols

<code>_APICore__inactive_database_management()</code> (<i>PyFunceble.core.api.APICore</i> method), 166	<code>_DNSLookup__request_ip()</code> (<i>PyFunceble.lookup.dns.DNSLookup</i> method), 192
<code>_AdBlock__decode_v1()</code> (<i>PyFunceble.converter.adblock.AdBlock</i> method), 163	<code>_DNSLookup__request_not_ip()</code> (<i>PyFunceble.lookup.dns.DNSLookup</i> method), 192
<code>_AdBlock__decode_v2()</code> (<i>PyFunceble.converter.adblock.AdBlock</i> method), 163	<code>_DomainAndIp__gather()</code> (<i>PyFunceble.status.availability.domain_and_ip.DomainAndIp</i> method), 208
<code>_AdBlock__decode_v3()</code> (<i>PyFunceble.converter.adblock.AdBlock</i> method), 163	<code>_DomainAndIp__gather()</code> (<i>PyFunceble.status.reputation.domain_and_ip.DomainAndIp</i> method), 210
<code>_AdBlock__decode_v4()</code> (<i>PyFunceble.converter.adblock.AdBlock</i> method), 163	<code>_DomainAndIp__gather()</code> (<i>PyFunceble.status.syntax.domain_and_ip.DomainAndIp</i> method), 210
<code>_AutoContinue__generate_complements()</code> (<i>PyFunceble.engine.auto_continue.AutoContinue</i> method), 176	<code>_DomainAndIp__gather_expiration_date()</code> (<i>PyFunceble.status.availability.domain_and_ip.DomainAndIp</i> method), 208
<code>_AutoContinue__get_or_generate_complements()</code> (<i>PyFunceble.engine.auto_continue.AutoContinue</i> method), 176	<code>_DomainAndIp__gather_extra_rules()</code> (<i>PyFunceble.status.availability.domain_and_ip.DomainAndIp</i> method), 208
<code>_AutoContinue__get_or_generate_complements_mysql()</code> (<i>PyFunceble.engine.auto_continue.AutoContinue</i> method), 176	<code>_ExpirationDate__extract_it()</code> (<i>PyFunceble.extractor.expiration_date.ExpirationDate</i> method), 182
<code>_CLICore__check_deprecated()</code> (<i>PyFunceble.core.cli.CLICore</i> class method), 167	<code>_ExpirationDate__format_it()</code> (<i>PyFunceble.extractor.expiration_date.ExpirationDate</i> method), 182
<code>_CLICore__check_force_update()</code> (<i>PyFunceble.core.cli.CLICore</i> class method), 167	<code>_ExpirationDate__format_management()</code> (<i>PyFunceble.extractor.expiration_date.ExpirationDate</i> method), 182
<code>_CLICore__print_messages()</code> (<i>PyFunceble.core.cli.CLICore</i> class method), 167	<code>_ExtraRules__blogspot()</code> (<i>PyFunceble.status.availability.extra_rules.ExtraRules</i> method), 208
<code>_DNSLookup__get_dns_servers_from()</code> (<i>PyFunceble.lookup.dns.DNSLookup</i> method), 191	<code>_ExtraRules__handle_http_code()</code> (<i>PyFunceble.status.availability.extra_rules.ExtraRules</i> method), 209
<code>_DNSLookup__get_resolver()</code> (<i>PyFunceble.lookup.dns.DNSLookup</i> class method), 191	<code>_ExtraRules__handle_ip_range()</code> (<i>PyFunceble.status.availability.extra_rules.ExtraRules</i> method), 209
<code>_DNSLookup__get_server_and_port_from()</code> (<i>PyFunceble.lookup.dns.DNSLookup</i> method), 191	<code>_ExtraRules__handle_potentially_inactive()</code>
<code>_DNSLookup__request_complete_not_ip()</code> (<i>PyFunceble.lookup.dns.DNSLookup</i> method), 192	

(*PyFunceble.status.availability.extra_rules.ExtraRules* method), 189
_Merge__list() (*PyFunceble.helpers.Merge* method), 209
_ExtraRules__handle_potentially_up() (*PyFunceble.status.availability.extra_rules.ExtraRules* method), 209
_ExtraRules__handle_reputation() (*PyFunceble.status.availability.extra_rules.ExtraRules* method), 209
_ExtraRules__http_status_code_up() (*PyFunceble.status.availability.extra_rules.ExtraRules* class method), 209
_ExtraRules__special_down() (*PyFunceble.status.availability.extra_rules.ExtraRules* class method), 209
_ExtraRules__special_down_404() (*PyFunceble.status.availability.extra_rules.ExtraRules* method), 209
_ExtraRules__special_up() (*PyFunceble.status.availability.extra_rules.ExtraRules* class method), 209
_ExtraRules__wordpress_dot_com() (*PyFunceble.status.availability.extra_rules.ExtraRules* method), 209
_FileCore__run_single_test() (*PyFunceble.core.file.FileCore* method), 168
_FileCore__test_line() (*PyFunceble.core.file.FileCore* method), 168
_Generate__get_info_files_destinations() (*PyFunceble.output.generate.Generate* method), 202
_Generate__info_files_authorization() (*PyFunceble.output.generate.Generate* method), 202
_HostAdapter__resolve_with_cache() (*PyFunceble.lookup.requests.HostAdapter* method), 197
_HostSSLAdapter__resolve_with_cache() (*PyFunceble.lookup.requests.HostSSLAdapter* method), 196
_IPv4Reputation__is_present() (*PyFunceble.lookup.ipv4_reputation.IPv4Reputation* method), 198
_Iana__get_content() (*PyFunceble.lookup.iana.Iana* method), 196
_InactiveDB__execute_query() (*PyFunceble.database.inactive.InactiveDB* method), 171
_Logger__get_handler() (*PyFunceble.engine.logger.Logger* method), 177
_Logger__init_loggers() (*PyFunceble.engine.logger.Logger* method), 177
_Logger__set_output_directory() (*PyFunceble.engine.logger.Logger* method), 177
_Merge__dict() (*PyFunceble.helpers.Merge* method), 189
_Merge__list() (*PyFunceble.helpers.Merge* method), 209
_MultiprocessCore__check_exception() (*PyFunceble.core.multiprocess.MultiprocessCore* method), 169
_MultiprocessCore__merge_processes_data() (*PyFunceble.core.multiprocess.MultiprocessCore* method), 169
_MultiprocessCore__run_multiprocess_test() (*PyFunceble.core.multiprocess.MultiprocessCore* method), 169
_MultiprocessCore__start_process() (*PyFunceble.core.multiprocess.MultiprocessCore* method), 169
_Preset__are_we_allowed_to_overwrite() (*PyFunceble.config.preset.Preset* class method), 161
_Prints__get_print_size() (*PyFunceble.output.prints.Prints* method), 205
_PublicSuffix__get_content() (*PyFunceble.lookup.publicsuffix.PublicSuffix* method), 196
_SimpleCore__save_in_database() (*PyFunceble.core.simple.SimpleCore* method), 170
_Url__gather() (*PyFunceble.status.availability.url.Url* method), 210
_Url__gather() (*PyFunceble.status.reputation.url.Url* method), 210
_Url__gather() (*PyFunceble.status.syntax.url.Url* method), 210
_analytic_host_file_directory() (*PyFunceble.output.generate.Generate* method), 203
_colorify() (*PyFunceble.output.prints.Prints* method), 205
_create_directory() (*PyFunceble.output.constructor.Constructor* class method), 201
_decode_output() (*PyFunceble.helpers.Command* method), 183
_do_not_produce_file() (*PyFunceble.output.generate.Generate* method), 203
_get_content() (*PyFunceble.output.logs.Logs* class method), 203
_get_it() (*PyFunceble.lookup.http_code.HTTPCode* method), 195
_get_structure() (*PyFunceble.output.constructor.Constructor* method), 201
_json_print() (*PyFunceble.output.prints.Prints* method), 205
_merge() (*PyFunceble.database.inactive.InactiveDB*

method), 171
 _prints_status_screen() (PyFunceble.output.generate.Generate method), 203
 _restore_replace() (PyFunceble.output.constructor.Constructor method), 201
 _size_from_header() (PyFunceble.output.prints.Prints class method), 205
 _update_structure_from_config() (PyFunceble.output.constructor.Constructor method), 201
 _write_content() (PyFunceble.output.logs.Logs class method), 203

A

a_record() (PyFunceble.lookup.dns.DNSLookup method), 192
 aaaa_record() (PyFunceble.lookup.dns.DNSLookup method), 192
 AdBlock (class in PyFunceble.converter.adblock), 163
 add() (PyFunceble.database.inactive.InactiveDB method), 171
 add() (PyFunceble.database.whois.WhoisDB method), 172
 add() (PyFunceble.engine.auto_continue.AutoContinue method), 176
 almost_everything() (PyFunceble.output.clean.Clean method), 201
 analytic_file() (PyFunceble.output.generate.Generate method), 203
 api() (PyFunceble.config.preset.Preset method), 161
 APICore (class in PyFunceble.core.api), 166
 are_tables_present() (PyFunceble.engine.mysql.MySQL method), 179
 authorization() (PyFunceble.cli.execution_time.ExecutionTime class method), 159
 authorization() (PyFunceble.database.inactive.InactiveDB class method), 171
 authorization() (PyFunceble.database.whois.WhoisDB class method), 172
 authorization() (PyFunceble.engine.auto_continue.AutoContinue class method), 176
 authorization() (PyFunceble.engine.logger.Logger method), 177
 authorization() (PyFunceble.engine.mining.Mining class method), 178
 authorization() (PyFunceble.engine.mysql.MySQL class method), 179

AutoContinue (class in PyFunceble.engine.auto_continue), 176
 AutoSave (class in PyFunceble.engine.auto_save), 177
 availability() (PyFunceble.core.api.APICore method), 166

B

backup() (PyFunceble.output.constructor.Constructor method), 201
 before_header() (PyFunceble.output.prints.Prints method), 205

C

calculate() (PyFunceble.cli.execution_time.ExecutionTime class method), 159
 calculate() (PyFunceble.output.percentage.Percentage class method), 204
 Check (class in PyFunceble.check), 149
 Clean (class in PyFunceble.output.clean), 200
 clean() (PyFunceble.database.inactive.InactiveDB method), 171
 clean() (PyFunceble.engine.auto_continue.AutoContinue method), 176
 cleanup() (PyFunceble.core.file.FileCore method), 168
 CLICore (class in PyFunceble.core.cli), 167
 cname_record() (PyFunceble.lookup.dns.DNSLookup method), 192
 colorify_logo() (PyFunceble.core.cli.CLICore class method), 167
 Command (class in PyFunceble.helpers), 183
 compare() (PyFunceble.abstracts.package.Version class method), 155
 compare_version_and_print_messages() (PyFunceble.core.cli.CLICore class method), 167
 complements() (PyFunceble.config.preset.Preset method), 161
 complements_file() (PyFunceble.output.generate.Generate method), 203
 ConfigDownloader (class in PyFunceble.downloader.config), 174
 CONFIGURATION_FILENAME (PyFunceble.abstracts.infrastructure.Infrastructure attribute), 154
 ConfigurationFileNotFound, 152
 construct_and_get_shadow_file() (PyFunceble.core.file.FileCore method), 168
 construct_and_get_shadow_file() (PyFunceble.core.multiprocess.MultiprocessCore method), 170

Constructor (class in *PyFunceble.output.constructor*), 201

ConverterBase (class in *PyFunceble.converter.base*), 163

cooldown_time() (*PyFunceble.config.preset.Preset* method), 161

copy() (*PyFunceble.helpers.File* method), 187

count() (*PyFunceble.output.percentage.Percentage* method), 204

create() (*PyFunceble.helpers.Directory* method), 186

create_tables_and_apply_patches() (*PyFunceble.engine.mysql.MySQL* method), 179

critical() (*PyFunceble.engine.logger.Logger* method), 177

custom_format() (*PyFunceble.helpers.List* method), 189

D

data() (*PyFunceble.helpers.Hash* method), 188

data() (*PyFunceble.output.prints.Prints* method), 205

data_constructor() (*PyFunceble.output.prints.Prints* method), 205

databases_to_delete() (*PyFunceble.output.clean.Clean* class method), 201

datetime() (*PyFunceble.database.inactive.InactiveDB* class method), 171

db_types() (*PyFunceble.config.preset.Preset* class method), 161

DBTypeDownloader (class in *PyFunceble.downloader.db_type*), 175

debug() (*PyFunceble.engine.logger.Logger* method), 177

DEFAULT_CONFIGURATION_FILENAME (*PyFunceble.abstracts.infrastructure.Infrastructure* attribute), 154

delete() (*PyFunceble.helpers.Directory* method), 186

delete() (*PyFunceble.helpers.File* method), 187

delete() (*PyFunceble.lookup.requests.Requests* method), 198

delete_unneeded() (*PyFunceble.output.constructor.Constructor* method), 201

Dict (class in *PyFunceble.helpers*), 184

Digit2Digits (class in *PyFunceble.converter.digit2digits*), 164

Directory (class in *PyFunceble.helpers*), 186

DirectoryStructureDownloader (class in *PyFunceble.downloader.directory_structure*), 175

disable() (*PyFunceble.config.preset.Preset* class method), 161

dispatch_file_test() (*PyFunceble.cli.dispatcher.Dispatcher* class method), 158

dispatch_url_file_test() (*PyFunceble.cli.dispatcher.Dispatcher* method), 158

Dispatcher (class in *PyFunceble.cli.dispatcher*), 157

dname_record() (*PyFunceble.lookup.dns.DNSLookup* method), 193

dns_lookup() (in module *PyFunceble*), 110

dns_lookup_over_tcp() (*PyFunceble.config.preset.Preset* class method), 161

dns_nameserver() (*PyFunceble.config.preset.Preset* class method), 161

DNSLookup (class in *PyFunceble.lookup.dns*), 191

domain() (*PyFunceble.core.simple.SimpleCore* method), 170

domain_and_ip() (*PyFunceble.core.api.APICore* method), 166

domain_syntax() (*PyFunceble.core.api.APICore* method), 166

DomainAndIp (class in *PyFunceble.status.availability.domain_and_ip*), 208

DomainAndIp (class in *PyFunceble.status.reputation.domain_and_ip*), 210

DomainAndIp (class in *PyFunceble.status.syntax.domain_and_ip*), 210

DOWN_FILENAME (*PyFunceble.abstracts.infrastructure.Infrastructure* attribute), 154

Download (class in *PyFunceble.helpers*), 186

download_link() (*PyFunceble.core.file.FileCore* class method), 168

DownloaderBase (class in *PyFunceble.downloader.base*), 174

DOWNTIME_INDEX (*PyFunceble.downloader.base.DownloaderBase* attribute), 174

E

enable() (*PyFunceble.config.preset.Preset* class method), 161

ENV_FILENAME (*PyFunceble.abstracts.infrastructure.Infrastructure* attribute), 154

error() (*PyFunceble.engine.logger.Logger* method), 177

errors (*PyFunceble.engine.mysql.MySQL* attribute), 179

exception (*PyFunceble.core.multiprocess.OurProcessWrapper* attribute), 169

exception() (*PyFunceble.engine.logger.Logger* method), 178

exceptions (*PyFunceble.lookup.requests.Requests* attribute), 198

execute() (*PyFunceble.helpers.Command* method), 183

ExecutionTime (class in PyFunceble.cli.execution_time), 159
 exists() (PyFunceble.helpers.Directory method), 186
 exists() (PyFunceble.helpers.File method), 187
 expiration_date() (PyFunceble.output.logs.Logs method), 204
 ExpirationDate (class in PyFunceble.extractor.expiration_date), 182
 extract_base() (PyFunceble.converter.adblock.Adblock method), 163
 extract_from_options() (PyFunceble.converter.adblock.Adblock method), 163
 ExtractorBase (class in PyFunceble.extractor.base), 181
 ExtraRules (class in PyFunceble.status.availability.extra_rules), 208

F

fatal() (PyFunceble.engine.logger.Logger method), 178
 File (class in PyFunceble.converter.file), 164
 File (class in PyFunceble.helpers), 187
 file() (PyFunceble.helpers.Hash method), 188
 file_to_delete() (PyFunceble.output.clean.Clean class method), 201
 file_url() (PyFunceble.config.preset.Preset method), 161
 FileCore (class in PyFunceble.core.file), 168
 fix_path() (PyFunceble.helpers.Directory method), 186
 format() (PyFunceble.helpers.List method), 189
 format_decoded() (PyFunceble.converter.adblock.Adblock method), 164
 format_execution_time() (PyFunceble.cli.execution_time.ExecutionTime method), 159
 format_to_apply (PyFunceble.engine.logger.Logger attribute), 178
 from_json() (PyFunceble.helpers.Dict class method), 184
 from_json_file() (PyFunceble.helpers.Dict class method), 184
 from_yaml() (PyFunceble.helpers.Dict class method), 184
 from_yaml_file() (PyFunceble.helpers.Dict class method), 184

G

gather_http_status_code() (PyFunceble.status.gatherer_base.GathererBase method), 207
 GathererBase (class in PyFunceble.status.gatherer_base), 207
 Generate (class in PyFunceble.output.generate), 202
 generate_complement_status_file() (PyFunceble.core.file.FileCore method), 168
 generate_files() (PyFunceble.core.file.FileCore method), 168
 generate_files_of_status() (PyFunceble.core.file.FileCore method), 168
 get() (PyFunceble.abstracts.platform.Platform class method), 155
 get() (PyFunceble.lookup.http_code.HTTPCode method), 195
 get() (PyFunceble.lookup.referer.Referer method), 196
 get() (PyFunceble.lookup.requests.Requests method), 198
 get() (PyFunceble.status.gatherer_base.GathererBase method), 207
 get() (PyFunceble.status.status.Status method), 208
 get_addr_info() (PyFunceble.lookup.dns.DNSLookup class method), 193
 get_all_downtimes() (PyFunceble.downloader.base.DownloaderBase method), 174
 get_already_tested() (PyFunceble.database.inactive.InactiveDB method), 172
 get_already_tested() (PyFunceble.engine.auto_continue.AutoContinue method), 176
 get_complements() (in module PyFunceble), 110
 get_complements() (PyFunceble.core.file.FileCore class method), 168
 get_converted() (PyFunceble.converter.adblock.Adblock method), 164
 get_converted() (PyFunceble.converter.base.ConverterBase method), 163
 get_converted() (PyFunceble.converter.file.File method), 164
 get_current() (PyFunceble.helpers.Directory class method), 186
 get_current_ci() (PyFunceble.engine.auto_save.AutoSave class method), 177
 get_downtime() (PyFunceble.downloader.base.DownloaderBase method), 174
 get_expiration_date() (PyFunceble.database.whois.WhoisDB method), 172
 get_extracted() (PyFunceble.extractor.base.ExtractorBase method),

- 181
- `get_history()` (*PyFunceble.engine.mining.Mining class method*), 178
- `get_host_by_addr()` (*PyFunceble.lookup.dns.DNSLookup class method*), 193
- `get_int_cast_type()` (*PyFunceble.engine.mysql.MySQL class method*), 179
- `get_matching_list()` (*PyFunceble.helpers.Regex method*), 190
- `get_not_matching_list()` (*PyFunceble.helpers.Regex method*), 190
- `get_or_generate_complements()` (*PyFunceble.engine.auto_continue.AutoContinue method*), 176
- `get_origin_info()` (*PyFunceble.engine.logger.Logger class method*), 178
- `get_simple_coloration()` (*PyFunceble.core.cli.CLICore class method*), 167
- `get_size()` (*PyFunceble.helpers.File method*), 187
- `get_subjects()` (*PyFunceble.core.file.FileCore class method*), 168
- `get_table_name()` (*PyFunceble.database.inactive.InactiveDB class method*), 172
- `get_table_name()` (*PyFunceble.database.whois.WhoisDB class method*), 172
- `get_table_name()` (*PyFunceble.engine.auto_continue.AutoContinue class method*), 176
- `get_table_name()` (*PyFunceble.engine.mining.Mining class method*), 178
- `get_to_clean()` (*PyFunceble.database.inactive.InactiveDB method*), 172
- `get_to_retest()` (*PyFunceble.database.inactive.InactiveDB method*), 172
- `get_upstream_version_file()` (*PyFunceble.core.cli.CLICore class method*), 167
- GitEmailNotFound, 152
- GitHubTokenNotFound, 152
- GitLabTokenNotFound, 152
- GitNameNotFound, 152
- ## H
- `handle()` (*PyFunceble.status.availability.extra_rules.ExtraRules method*), 210
- `has_same_keys_as()` (*PyFunceble.helpers.Dict method*), 184
- Hash (*class in PyFunceble.helpers*), 188
- `head()` (*PyFunceble.lookup.requests.Requests method*), 198
- `header()` (*PyFunceble.output.prints.Prints method*), 205
- `header_constructor()` (*PyFunceble.output.prints.Prints class method*), 206
- `hierarchical()` (*PyFunceble.engine.sort.Sort class method*), 180
- HostAdapter (*class in PyFunceble.lookup.requests*), 197
- HostSSLAdapter (*class in PyFunceble.lookup.requests*), 196
- HTTPCode (*class in PyFunceble.lookup.http_code*), 195
- ## I
- Iana (*class in PyFunceble.lookup.iana*), 196
- IANA_FILENAME (*PyFunceble.abstracts.infrastructure.Infrastructure attribute*), 154
- IANADownloader (*class in PyFunceble.downloader.iana*), 175
- `ignore_it()` (*PyFunceble.converter.adblock.Adblock class method*), 164
- InactiveDB (*class in PyFunceble.database.inactive*), 171
- `info()` (*PyFunceble.engine.logger.Logger method*), 178
- `info_files()` (*PyFunceble.output.generate.Generate method*), 203
- Infrastructure (*class in PyFunceble.abstracts.infrastructure*), 154
- `init()` (*PyFunceble.engine.logger.Logger method*), 178
- `init_all()` (*PyFunceble.config.preset.Preset method*), 161
- `init_post_connection()` (*PyFunceble.engine.mysql.MySQL method*), 179
- `init_pre_connection()` (*PyFunceble.engine.mysql.MySQL method*), 179
- `initiate()` (*PyFunceble.database.inactive.InactiveDB method*), 172
- InternalUrl (*class in PyFunceble.converter.internal_url*), 165
- `into()` (*PyFunceble.helpers.Merge method*), 189
- `ip_range_syntax()` (*PyFunceble.core.api.APICore method*), 166
- `ip_syntax()` (*PyFunceble.core.api.APICore method*), 166
- `ip_range_syntax()` (*PyFunceble.core.api.APICore method*), 166
- IPV4_REPUTATION_FILENAME (*PyFunceble.abstracts.infrastructure.Infrastructure*

- attribute*), 154
- `ipv4_syntax()` (*PyFunceble.core.api.APICore method*), 166
- `ipv4_syntax_check()` (*in module PyFunceble*), 110
- `IPv4Reputation` (*class in PyFunceble.lookup.ipv4_reputation*), 198
- `IPv4ReputationDownloader` (*class in PyFunceble.downloader.ipv4_reputation*), 175
- `ipv6_range_syntax()` (*PyFunceble.core.api.APICore method*), 166
- `ipv6_syntax()` (*PyFunceble.core.api.APICore method*), 166
- `is_domain()` (*in module PyFunceble*), 111
- `is_domain()` (*PyFunceble.check.Check method*), 150
- `is_domain_malicious()` (*in module PyFunceble*), 111
- `is_downtime_set()` (*PyFunceble.downloader.base.DownloaderBase method*), 174
- `is_empty()` (*PyFunceble.engine.auto_continue.AutoContinue method*), 176
- `is_empty()` (*PyFunceble.helpers.File method*), 187
- `is_ip()` (*in module PyFunceble*), 111
- `is_ip()` (*PyFunceble.check.Check method*), 150
- `is_ip_range()` (*in module PyFunceble*), 111
- `is_ip_range()` (*PyFunceble.check.Check method*), 150
- `is_ipv4()` (*in module PyFunceble*), 111
- `is_ipv4()` (*PyFunceble.check.Check method*), 150
- `is_ipv4_malicious()` (*in module PyFunceble*), 111
- `is_ipv4_range()` (*in module PyFunceble*), 111
- `is_ipv4_range()` (*PyFunceble.check.Check method*), 150
- `is_ipv6()` (*in module PyFunceble*), 111
- `is_ipv6()` (*PyFunceble.check.Check method*), 150
- `is_ipv6_range()` (*PyFunceble.check.Check method*), 150
- `is_last_download_expired()` (*PyFunceble.downloader.base.DownloaderBase method*), 174
- `is_local_cloned()` (*PyFunceble.abstracts.package.Version class method*), 156
- `is_local_dev()` (*PyFunceble.abstracts.package.Version class method*), 156
- `is_mac_os()` (*PyFunceble.abstracts.platform.Platform class method*), 155
- `is_record_present_in_result()` (*PyFunceble.lookup.dns.DNSLookup class method*), 193
- `is_reserved_ip()` (*PyFunceble.check.Check method*), 150
- `is_reserved_ipv4()` (*PyFunceble.check.Check method*), 150
- `is_reserved_ipv6()` (*PyFunceble.check.Check method*), 151
- `is_subdomain()` (*in module PyFunceble*), 112
- `is_subdomain()` (*PyFunceble.check.Check method*), 151
- `is_time_exceed()` (*PyFunceble.engine.auto_save.AutoSave method*), 177
- `is_time_older()` (*PyFunceble.database.whois.WhoisDB method*), 172
- `is_unix()` (*PyFunceble.abstracts.platform.Platform class method*), 155
- `is_url()` (*in module PyFunceble*), 112
- `is_url()` (*PyFunceble.check.Check method*), 151
- `is_url_malicious()` (*in module PyFunceble*), 112
- `is_windows()` (*PyFunceble.abstracts.platform.Platform class method*), 155
- ## L
- `List` (*class in PyFunceble.helpers*), 188
- `list_of_mined()` (*PyFunceble.engine.mining.Mining method*), 178
- `literally_compare()` (*PyFunceble.abstracts.package.Version class method*), 156
- `load()` (*PyFunceble.database.inactive.InactiveDB method*), 172
- `load()` (*PyFunceble.database.whois.WhoisDB method*), 173
- `load()` (*PyFunceble.engine.auto_continue.AutoContinue method*), 177
- `load()` (*PyFunceble.engine.mining.Mining method*), 179
- `load_config()` (*in module PyFunceble*), 112
- `log()` (*PyFunceble.output.percentage.Percentage method*), 204
- `Logger` (*class in PyFunceble.engine.logger*), 177
- `Logs` (*class in PyFunceble.output.logs*), 203
- `logs_sharing()` (*PyFunceble.core.cli.CLICore class method*), 167
- ## M
- `MAC` (*PyFunceble.abstracts.platform.Platform attribute*), 155
- `match()` (*PyFunceble.helpers.Regex method*), 190
- `maximal_processes()` (*PyFunceble.config.preset.Preset class method*), 161
- `Merge` (*class in PyFunceble.helpers*), 189

`merge()` (*PyFunceble.database.whois.WhoisDB class method*), 173
`mine()` (*PyFunceble.engine.mining.Mining method*), 179
`Mining` (*class in PyFunceble.engine.mining*), 178
`move()` (*PyFunceble.helpers.File method*), 187
`multiprocess()` (*PyFunceble.config.preset.Preset method*), 161
`multiprocess_merging_mode()` (*PyFunceble.config.preset.Preset class method*), 161
`MultiprocessCore` (*class in PyFunceble.core.multiprocess*), 169
`mx_record()` (*PyFunceble.lookup.dns.DNSLookup method*), 193
`MySQL` (*class in PyFunceble.engine.mysql*), 179

N

`NAME` (*PyFunceble.abstracts.package.Package attribute*), 155
`NoConversionMade`, 152
`NoDownloadDestinationGiven`, 152
`NoDownloadLinkGiven`, 152
`NoExtractionMade`, 152
`NoInternetConnection`, 152
`ns_record()` (*PyFunceble.lookup.dns.DNSLookup method*), 194

O

`options()` (*PyFunceble.lookup.requests.Requests method*), 198
`OurProcessWrapper` (*class in PyFunceble.core.multiprocess*), 169

P

`Package` (*class in PyFunceble.abstracts.package*), 155
`parse_env_file()` (*PyFunceble.engine.mysql.MySQL class method*), 180
`parse_mysql_sql_file()` (*PyFunceble.engine.mysql.MySQL class method*), 180
`patch()` (*PyFunceble.lookup.requests.Requests method*), 198
`Percentage` (*class in PyFunceble.output.percentage*), 204
`Platform` (*class in PyFunceble.abstracts.platform*), 155
`PleaseUpdatePyFunceble`, 152
`post()` (*PyFunceble.lookup.requests.Requests method*), 199
`post_test_treatment()` (*PyFunceble.core.file.FileCore method*), 168
`Preset` (*class in PyFunceble.config.preset*), 161

`print_header()` (*PyFunceble.core.cli.CLICore class method*), 167
`print_nothing_to_test()` (*PyFunceble.core.cli.CLICore class method*), 167
`Prints` (*class in PyFunceble.output.prints*), 204
`prints_status_file()` (*PyFunceble.output.generate.Generate method*), 203
`process()` (*PyFunceble.downloader.base.DownloaderBase method*), 174
`process()` (*PyFunceble.engine.auto_save.AutoSave method*), 177
`PROD_CONFIG_LINK` (*PyFunceble.abstracts.infrastructure.Infrastructure attribute*), 154
`ptr_record()` (*PyFunceble.lookup.dns.DNSLookup method*), 194
`PublicSuffix` (*class in PyFunceble.lookup.publicsuffix*), 196
`PublicSuffixDownloader` (*class in PyFunceble.downloader.publicsuffix*), 175
`put()` (*PyFunceble.lookup.requests.Requests method*), 199
`PyFunceble` (*module*), 109
`PyFunceble.abstracts` (*module*), 153
`PyFunceble.check` (*module*), 149
`PyFunceble.cli` (*module*), 157
`PyFunceble.config` (*module*), 160
`PyFunceble.converter` (*module*), 162
`PyFunceble.core` (*module*), 165
`PyFunceble.database` (*module*), 170
`PyFunceble.downloader` (*module*), 173
`PyFunceble.engine` (*module*), 175
`PyFunceble.exceptions` (*module*), 151
`PyFunceble.extractor` (*module*), 181
`PyFunceble.helpers` (*module*), 182
`PyFunceble.lookup` (*module*), 190
`PyFunceble.output` (*module*), 200
`PyFunceble.status` (*module*), 206
`PyFuncebleException`, 152
`PyFuncebleExternalException`, 153
`PyFuncebleInternalException`, 153

R

`read()` (*PyFunceble.helpers.File method*), 188
`REDOWNLOAD_AFTER` (*PyFunceble.downloader.base.DownloaderBase attribute*), 174
`Referer` (*class in PyFunceble.lookup.referer*), 196
`referer_not_found()` (*PyFunceble.output.logs.Logs method*), 204
`Regex` (*class in PyFunceble.helpers*), 189
`remove()` (*PyFunceble.database.inactive.InactiveDB method*), 172

- [remove\(\) \(PyFunceble.engine.mining.Mining method\), 179](#)
[remove_ignored\(\) \(PyFunceble.converter.adblock.Adblock method\), 164](#)
[remove_key\(\) \(PyFunceble.helpers.Dict method\), 184](#)
[rename_key\(\) \(PyFunceble.helpers.Dict method\), 185](#)
[replace_match\(\) \(PyFunceble.helpers.Regex method\), 190](#)
[REPO_LINK \(PyFunceble.abstracts.infrastructure.Infrastructure attribute\), 154](#)
[reputation\(\) \(PyFunceble.core.api.APICore method\), 166](#)
[reputation_data\(\) \(PyFunceble.config.preset.Preset method\), 161](#)
[request\(\) \(PyFunceble.lookup.dns.DNSLookup method\), 194](#)
[request\(\) \(PyFunceble.lookup.whois.WhoisLookup method\), 199](#)
[Requests \(class in PyFunceble.lookup.requests\), 198](#)
[reset_counters\(\) \(PyFunceble.config.preset.Preset class method\), 161](#)
[resolve\(\) \(PyFunceble.lookup.requests.HostSSLAdapter class method\), 196](#)
[restore\(\) \(PyFunceble.output.constructor.Constructor method\), 201](#)
[root_logger_format \(PyFunceble.engine.logger.Logger attribute\), 178](#)
[run\(\) \(PyFunceble.core.multiprocess.OurProcessWrapper method\), 169](#)
[run\(\) \(PyFunceble.helpers.Command method\), 183](#)
[run_test\(\) \(PyFunceble.core.file.FileCore method\), 169](#)
[run_test\(\) \(PyFunceble.core.multiprocess.MultiprocessCore method\), 170](#)
[run_to_stdout\(\) \(PyFunceble.helpers.Command method\), 183](#)
- ## S
- [save\(\) \(PyFunceble.cli.execution_time.ExecutionTime method\), 159](#)
[save\(\) \(PyFunceble.database.inactive.InactiveDB method\), 172](#)
[save\(\) \(PyFunceble.database.whois.WhoisDB method\), 173](#)
[save\(\) \(PyFunceble.engine.auto_continue.AutoContinue method\), 177](#)
[save\(\) \(PyFunceble.engine.mining.Mining method\), 179](#)
[save_downtimes\(\) \(PyFunceble.downloader.base.DownloaderBase method\), 174](#)
[save_into_database\(\) \(PyFunceble.core.cli.CLICore class method\), 168](#)
[save_to_env_file\(\) \(PyFunceble.engine.mysql.MySQL method\), 180](#)
[send\(\) \(PyFunceble.lookup.requests.HostAdapter method\), 197](#)
[send\(\) \(PyFunceble.lookup.requests.HostSSLAdapter method\), 197](#)
[set_starting_time\(\) \(PyFunceble.cli.execution_time.ExecutionTime class method\), 159](#)
[set_stopping_time\(\) \(PyFunceble.cli.execution_time.ExecutionTime class method\), 159](#)
[should_be_ignored\(\) \(PyFunceble.core.file.FileCore class method\), 169](#)
[simple_domain\(\) \(PyFunceble.config.preset.Preset method\), 161](#)
[simple_url\(\) \(PyFunceble.config.preset.Preset method\), 161](#)
[SimpleCore \(class in PyFunceble.core.simple\), 170](#)
[Sort \(class in PyFunceble.engine.sort\), 180](#)
[sort_generated_files\(\) \(PyFunceble.core.cli.CLICore class method\), 168](#)
[SPECIAL_USE_DOMAIN_NAMES_EXTENSIONS \(PyFunceble.check.Check attribute\), 149](#)
[split_versions\(\) \(PyFunceble.abstracts.package.Version class method\), 156](#)
[standard\(\) \(PyFunceble.engine.sort.Sort class method\), 180](#)
[Status \(class in PyFunceble.status.status\), 207](#)
[status_file\(\) \(PyFunceble.output.generate.Generate method\), 203](#)
[stay_safe\(\) \(PyFunceble.core.cli.CLICore class method\), 168](#)
[subdomain_syntax\(\) \(PyFunceble.core.api.APICore method\), 167](#)
[switch\(\) \(PyFunceble.config.preset.Preset class method\), 161](#)
[syntax\(\) \(PyFunceble.core.api.APICore method\), 167](#)
[syntax_check\(\) \(in module PyFunceble\), 112](#)
[syntax_test\(\) \(PyFunceble.config.preset.Preset method\), 162](#)
- ## T
- [test\(\) \(in module PyFunceble\), 112](#)
[test\(\) \(PyFunceble.core.file.FileCore method\), 169](#)

`test()` (*PyFunceble.core.multiprocess.MultiprocessCore method*), 170

`test()` (*PyFunceble.core.simple.SimpleCore method*), 170

`text()` (*PyFunceble.helpers.Download method*), 186

`timeout()` (*PyFunceble.config.preset.Preset class method*), 162

`to_json()` (*PyFunceble.helpers.Dict method*), 185

`to_json_file()` (*PyFunceble.helpers.Dict method*), 185

`to_right_url()` (*PyFunceble.converter.internal_url.InternalUrl method*), 165

`to_yaml()` (*PyFunceble.helpers.Dict method*), 185

`to_yaml_file()` (*PyFunceble.helpers.Dict method*), 185

`tool()` (*in module PyFunceble.cli*), 157

`txt_record()` (*PyFunceble.lookup.dns.DNSLookup method*), 195

U

`unified_file()` (*PyFunceble.output.generate.Generate method*), 203

UNIX (*PyFunceble.abstracts.platform.Platform attribute*), 155

UnknownSubject, 153

`update_counters()` (*PyFunceble.engine.auto_continue.AutoContinue method*), 177

`update_downtime()` (*PyFunceble.downloader.base.DownloaderBase method*), 174

`update_lifetime()` (*PyFunceble.lookup.dns.DNSLookup method*), 195

`update_nameserver()` (*PyFunceble.lookup.dns.DNSLookup method*), 195

Url (*class in PyFunceble.status.availability.url*), 210

Url (*class in PyFunceble.status.reputation.url*), 210

Url (*class in PyFunceble.status.syntax.url*), 210

`url()` (*PyFunceble.core.api.APICore method*), 167

`url()` (*PyFunceble.core.simple.SimpleCore method*), 170

`url_syntax()` (*PyFunceble.core.api.APICore method*), 167

`url_syntax_check()` (*in module PyFunceble*), 113

`url_test()` (*in module PyFunceble*), 113

USER_AGENT_FILENAME (*PyFunceble.abstracts.infrastructure.Infrastructure attribute*), 154

UserAgentBrowserNotFound, 153

UserAgentNotFound, 153

UserAgentPlatformNotFound, 153

V

Version (*class in PyFunceble.abstracts.package*), 155

VERSION (*PyFunceble.abstracts.package.Package attribute*), 155

W

`warning()` (*PyFunceble.engine.logger.Logger method*), 178

`whois()` (*in module PyFunceble*), 114

`whois()` (*PyFunceble.output.logs.Logs method*), 204

WhoisDB (*class in PyFunceble.database.whois*), 172

WhoisLookup (*class in PyFunceble.lookup.whois*), 199

WINDOWS (*PyFunceble.abstracts.platform.Platform attribute*), 155

`work_process()` (*PyFunceble.core.multiprocess.MultiprocessCore method*), 170

`write()` (*PyFunceble.helpers.File method*), 188

`write_in_shadow_file_if_needed()` (*PyFunceble.core.file.FileCore method*), 169

WrongParameterType, 153

WrongParameterValue, 153