
wm_metrics Documentation

Release 0.1

Jean-Frédéric, Pierre-Selim, Caroline

Mar 04, 2017

Contents

1	Modules	3
1.1	wm_metrics package	3
1.2	Main modules	3
1.2.1	fdc module	3
1.2.2	analyse_commons_dump module	3
1.2.3	cat2cohort module	4
1.2.4	categorisation_statistics module	5
1.2.5	commons_cat_metrics module	5
1.2.6	mw_util module	6
1.2.7	wmflabs_queries module	6
1.3	External services	6
1.3.1	glamorous module	6
1.3.2	mw_api module	7
1.3.3	traffic_statistics module	7
2	Authors	9
3	Indices and tables	11
	Python Module Index	13

wm_metrics is a set of metrics tools for Wikimedia program leaders.

- **Repo:** https://github.com/Commonists/wm_metrics
- **Webapp:** <http://tools.wmflabs.org/wm-metrics/>

wm_metrics package

wm_metrics - A set of metrics tools for Wikimedia program leaders.

Main modules

fdc module

analyse_commons_dump module

Analysing a Commons collection to retrieve fancy statistics.

class wm_metrics.analyse_commons_dump.**CommonsPage** (*title=None, revisions=None*)

Bases: object

Represent a page.

get_top_revision ()

Return the most recent CommonsRevision.

We assume the revisions list is ordered by time (which is the case when initialized with a dump)

class wm_metrics.analyse_commons_dump.**CommonsRevision** (*timestamp=None, user-name=None, wikitext=None*)

Bases: object

Representation of a Revision (timestamp + username + wikitext).

get_categories ()

Return the categories in the given revision.

is_valued_image ()

Return whether the given revision is a Valued Image.

class `wm_metrics.analyse_commons_dump.DumpMediaCollection`

Bases: `dict`

Representation of a `MediaCollection`, dump style.

categorisation_report ()

Return a text categorisation report.

Iterate over the pages of the media collection, get the top revision, and collects the categories in two Counters - one indexed by category and the other one by file.

get_differential (*start_date*, *end_date*)

Return a difference between two dates.

get_initial_state ()

Return a `Collection` in its initial state.

get_state (*target_datetime*)

Return a `Collection` at the time given.

get_valued_images ()

Return a list of valued images in the collection.

init_from_xml_dump (*xml_dump*)

Initialise the object using an XML dump.

simple_all_time_report ()

Return an activity text report since the beginning to now.

This report on the number of edits, editors and files touched between two given dates.

simple_diff_report (*start_date*, *end_date*)

Return an activity text report in a given timeframe.

This report on the number of edits, editors and files touched between two given dates.

`wm_metrics.analyse_commons_dump.get_categories_from_text` (*edit*)

Return the categories contained in a given wikitext.

`wm_metrics.analyse_commons_dump.handle_node` (*node*, *tag_name*)

Return the contents of a tag based on his given name inside of a given node.

`wm_metrics.analyse_commons_dump.main` ()

`wm_metrics.analyse_commons_dump.parse_xml_dump` (*xml_dump*)

Return a dictionary from the given dump.

A dictionary structured as follow: {page_id => { CommonsPage(title => "Some title"git

revisions => [CommonsRevision, ...] } }

`wm_metrics.analyse_commons_dump.timestamp_to_date` (*date*)

Return a datetime object representing the given MediaWiki timestamp.

cat2cohort module

Export a Wiki category into a cohort.

The aim of this script is to allow program leaders to export a category filled with User pages into a WikiMetrics cohort CSV file in order to perform their evaluation analysis.

Test: `python cat2cohort.py -l fr -c "Utilisateur participant au projet Afripédia"`

`wm_metrics.cat2cohort.api_url (lang)`
Return the URL of the API based on the language of Wikipedia.

`wm_metrics.cat2cohort.cat_to_cohort (language, category)`
Return the CSV cohort from the given category and language.

`wm_metrics.cat2cohort.list_users (mw, category, lang)`
List users from a wiki category and print lines of the cohort CSV.

`wm_metrics.cat2cohort.main ()`
Main function of the script cat2cohort.

categorisation_statistics module

Categorisation statistics.

`wm_metrics.categorisation_statistics.make_categorisation_report (all_categories, cate- gories_count_per_file)`

Compute statistics on the categorisation.

Return a text report on the categorisation.

commons_cat_metrics module

Metrics for FDC on an image category of Wikimedia Commons.

class `wm_metrics.commonscat_metrics.CommonsCatMetrics (category, period, cursor=None)`

Bases: object

Wrapper class for the Category Metrics

close ()
Close the MariaDB connection.

get_global_usage (main=False)
Get global usage metrics (total usages, nb of images used, nb of wiki) of files in categories.

Parameters main (boolean) – whether we only count for main namespaces.

get_nb_featured_files ()
Amount of files that are either FP, VI or QI on Wikimedia Commons.

get_nb_files ()
Amount of files uploaded on the period.

get_nb_files_alltime ()
Returns nb of files in category.

get_nb_uploaders ()
Amount of uploaders on the period.

get_pixel_count ()

make_report ()
Return a text report with all metrics.

mw_util module

mw_util.py Set of helper functions while dealing with MediaWiki.

str2cat Adds prefix Category if string doesn't have it.

`wm_metrics.mw_util.str2cat(category)`

Return a category name starting with Category.

wmflabs_queries module

wmflabs_queries.py regroups query builder functions in order to generate queries for wmflabs databases.

`wm_metrics.wmflabs_queries.count_featured_files_in_category()`

Count featured pictures in the category uploaded between timestamp t1 and t2.

`wm_metrics.wmflabs_queries.count_files_in_category()`

List all files in category uploaded between timestamp t1 and t2

`wm_metrics.wmflabs_queries.count_files_in_category_alltime()`

Count files in the category (without limit on upload date) at the time of the query.

`wm_metrics.wmflabs_queries.count_uploaders_in_category()`

Count distinct users that have uploaded a files that belongs to category between timestamp t1 and t2

`wm_metrics.wmflabs_queries.global_usage_count(main=False)`

Returns global usage query

Parameters

- **category** (*str*) – category name
- **main** (*bool*) – optional in order to account only for file used in main namespaces

`wm_metrics.wmflabs_queries.list_files_in_category(category, t1, t2)`

List all files in category uploaded between timestamp t1 and t2

`wm_metrics.wmflabs_queries.pixel_count()`

External services

glamorous module

A Glamorous parser to retrieve file usage among the wikimedia projects.

class `wm_metrics.glamorous.GlamorousParser(category)`

Bases: `HTMLParser.HTMLParser`, `object`

HTML parser glamorous

handle_data (*data*)

Parse data inside an HTML tag.

handle_endtag (*tag*)

Parse end of an HTML tag.

handle_starttag (*tag*, *attrs*)

Parse start of an HTML tag.

statistics()

Print GLAMorous statistics for the category.

`wm_metrics.glamorous.main()`

Main function of the script glamorous.py.

mw_api module

mw_api.py is a simple client to MediaWiki API.

class `wm_metrics.mw_api.MwApi` (*action, properties=None, format='json'*)

Bases: `object`

Access to API

class `wm_metrics.mw_api.MwApiQuery` (*properties=None, format='json'*)

Bases: `wm_metrics.mw_api.MwApi`

Query actions to the API.

exception `wm_metrics.mw_api.MwQueryError` (*value*)

Bases: `exceptions.Exception`

Exception raised when the client encounters a problem.

class `wm_metrics.mw_api.MwWiki` (*url_api='https://commons.wikimedia.org/w/api.php'*)

Bases: `object`

Wiki API

process_prop_query (*request, titles*)

Quick and dirty prop query support.

process_prop_query_results (*url_req, results*)

Process the result of a prop query.

process_query (*request, previous_result=None*)

Quick and dirty continue support for list query.

send_to_api (*request, debug=False*)

Send a request to mediawiki API.

Parameters

- **request** (`MwApi`) – Request to send.
- **debug** (`bool`) – if true, then just only return the string of the API request, otherwise return the result.

traffic_statistics module

Traffic statistics API to grok.

class `wm_metrics.traffic_statistics.Traffic` (*title, site*)

Bases: `object`

Wikipedia article statistics.

get_latest_traffic (*latest*)

Fetch the latest traffic statistics.

get_month_traffic (*year, month*)
Fetch the month traffic statistics.

CHAPTER 2

Authors

Jean-Frédéric, Pierre-Selim, Caroline

CHAPTER 3

Indices and tables

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

W

- `wm_metrics`, [3](#)
- `wm_metrics.analyse_commons_dump`, [3](#)
- `wm_metrics.cat2cohort`, [4](#)
- `wm_metrics.categorisation_statistics`, [5](#)
- `wm_metrics.common_cat_metrics`, [5](#)
- `wm_metrics.glamorous`, [6](#)
- `wm_metrics.mw_api`, [7](#)
- `wm_metrics.mw_util`, [6](#)
- `wm_metrics.traffic_statistics`, [7](#)
- `wm_metrics.wmflabs_queries`, [6](#)

A

api_url() (in module wm_metrics.cat2cohort), 4

C

cat_to_cohort() (in module wm_metrics.cat2cohort), 5

categorisation_report() (wm_metrics.analyse_commons_dump.DumpMediaCollection
method), 4

close() (wm_metrics.common_cat_metrics.CommonsCatMetrics,
method), 5

CommonsCatMetrics (class in
wm_metrics.common_cat_metrics), 5

CommonsPage (class in
wm_metrics.analyse_commons_dump), 3

CommonsRevision (class in
wm_metrics.analyse_commons_dump), 3

count_featured_files_in_category() (in module
wm_metrics.wmflabs_queries), 6

count_files_in_category() (in module
wm_metrics.wmflabs_queries), 6

count_files_in_category_alltime() (in module
wm_metrics.wmflabs_queries), 6

count_uploaders_in_category() (in module
wm_metrics.wmflabs_queries), 6

D

DumpMediaCollection (class in
wm_metrics.analyse_commons_dump), 3

G

get_categories() (wm_metrics.analyse_commons_dump.CommonsRevision
method), 3

get_categories_from_text() (in module
wm_metrics.analyse_commons_dump), 4

get_differential() (wm_metrics.analyse_commons_dump.DumpMediaCollection
method), 4

get_global_usage() (wm_metrics.common_cat_metrics.CommonsCatMetrics,
method), 5

get_initial_state() (wm_metrics.analyse_commons_dump.DumpMediaCollection
method), 4

get_latest_traffic() (wm_metrics.traffic_statistics.Traffic
method), 7

get_month_traffic() (wm_metrics.traffic_statistics.Traffic
method), 7

get_nb_featured_files() (wm_metrics.common_cat_metrics.CommonsCatMetrics,
method), 5

get_nb_files() (wm_metrics.common_cat_metrics.CommonsCatMetrics,
method), 5

get_nb_files_alltime() (wm_metrics.common_cat_metrics.CommonsCatMetrics,
method), 5

get_nb_uploaders() (wm_metrics.common_cat_metrics.CommonsCatMetrics,
method), 5

get_pixel_count() (wm_metrics.common_cat_metrics.CommonsCatMetrics,
method), 5

get_state() (wm_metrics.analyse_commons_dump.DumpMediaCollection
method), 4

get_top_revision() (wm_metrics.analyse_commons_dump.CommonsPage
method), 3

get_valued_images() (wm_metrics.analyse_commons_dump.DumpMediaCollection
method), 4

GlamorousParser (class in wm_metrics.glamorous), 6

global_usage_count() (in module
wm_metrics.wmflabs_queries), 6

H

handle_data() (wm_metrics.glamorous.GlamorousParser
method), 6

handle_endtag() (wm_metrics.glamorous.GlamorousParser
method), 6

handle_node() (in module
wm_metrics.analyse_commons_dump), 4

handle_starttag() (wm_metrics.glamorous.GlamorousParser
method), 6

init_from_xml_dump() (wm_metrics.analyse_commons_dump.DumpMediaCollection
method), 4

is_valued_image() (wm_metrics.analyse_commons_dump.CommonsRevision
method), 3

L

`list_files_in_category()` (in module `wm_metrics.wmflabs_queries`), 6

`list_users()` (in module `wm_metrics.cat2cohort`), 5

`wm_metrics.common_cat_metrics` (module), 5

`wm_metrics.glamorous` (module), 6

`wm_metrics.mw_api` (module), 7

`wm_metrics.mw_util` (module), 6

`wm_metrics.traffic_statistics` (module), 7

`wm_metrics.wmflabs_queries` (module), 6

M

`main()` (in module `wm_metrics.analyse_commons_dump`), 4

`main()` (in module `wm_metrics.cat2cohort`), 5

`main()` (in module `wm_metrics.glamorous`), 7

`make_categorisation_report()` (in module `wm_metrics.categorisation_statistics`), 5

`make_report()` (`wm_metrics.common_cat_metrics.CommonsCatMetrics` method), 5

`MwApi` (class in `wm_metrics.mw_api`), 7

`MwApiQuery` (class in `wm_metrics.mw_api`), 7

`MwQueryError`, 7

`MwWiki` (class in `wm_metrics.mw_api`), 7

P

`parse_xml_dump()` (in module `wm_metrics.analyse_commons_dump`), 4

`pixel_count()` (in module `wm_metrics.wmflabs_queries`), 6

`process_prop_query()` (`wm_metrics.mw_api.MwWiki` method), 7

`process_prop_query_results()` (`wm_metrics.mw_api.MwWiki` method), 7

`process_query()` (`wm_metrics.mw_api.MwWiki` method), 7

S

`send_to_api()` (`wm_metrics.mw_api.MwWiki` method), 7

`simple_all_time_report()` (`wm_metrics.analyse_commons_dump.DumpMediaCollection` method), 4

`simple_diff_report()` (`wm_metrics.analyse_commons_dump.DumpMediaCollection` method), 4

`statistics()` (`wm_metrics.glamorous.GlamorousParser` method), 6

`str2cat()` (in module `wm_metrics.mw_util`), 6

T

`timestamp_to_date()` (in module `wm_metrics.analyse_commons_dump`), 4

`Traffic` (class in `wm_metrics.traffic_statistics`), 7

W

`wm_metrics` (module), 3

`wm_metrics.analyse_commons_dump` (module), 3

`wm_metrics.cat2cohort` (module), 4

`wm_metrics.categorisation_statistics` (module), 5