# InfluxGraph Documentation

*Release 1.5.0.post1+1.g347c9b4.dirty*

**PK, Dieter Plaetinck**

**Nov 07, 2018**

# Contents


1 InfluxDB finder 3

2 Node Index Tree for Graphite metrics 5

3 Graphite template related functions 7

4 Utility functions and classes 9

5 Indices and tables 11

Python Module Index 13

Contents:

# CHAPTER 1

# InfluxDB finder

Graphite-Api storage finder for InfluxDB.

Read metric series from an InfluxDB database via a Graphite-API storage plugin compatible API.

**class** `influxgraph.classes.finder.`**`InfluxDBFinder`**(*config*)

Graphite-Api finder for InfluxDB.

Finds and fetches metric series from InfluxDB.

**`build_index`**(*data=None*, *separator='.'*)

Build new node tree index

**Parameters** **`data`** (*`list`*) – (Optional) data to use to build index

**`fetch_multi`**(*nodes*, *start_time*, *end_time*)

Fetch datapoints for all series between start and end times

**Parameters**

- **`nodes`** (list(`influxgraph.classes.InfluxDBLeafNode`)) – List of nodes to retrieve data for
- **`start_time`** – Start time of query
- **`end_time`** – End time of query

**`find_nodes`**(*query*)

Find and return nodes matching query

**Parameters** **`query`** (*`influxgraph.utils.Query`*) – Query to search for

**`get_all_series`**(*cache=True*, *offset=0*, *_data=None*, ***kwargs*)

Retrieve all series

**`get_all_series_list`**(*offset=0*, *_data=None*, **args*, ***kwargs*)

Retrieve all series for series loader

**`get_field_keys`**()

Get field keys for all measurements

**get_series**(*cache=True*, *offset=0*)

Retrieve series names from InfluxDB according to query pattern

**Parameters** **query** (`graphite_api.storage.FindQuery` compatible class) – Query to run to get series names

**load_index**()

Load index from file

**save_index**()

Save index to file

# CHAPTER 2

## Node Index Tree for Graphite metrics

Tree representation of Graphite metrics

**class** `influxgraph.classes.tree.`**`Node`**

Node class of a graphite metric

**static from_array**(*array*)

Load given parent node's children from array

**insert**(*paths*)

Insert path in this node's children

**is_leaf**()

Returns True/False depending on whether self is a LeafNode or not

**to_array**()

Return list of (name, children) items for this node's children

**class** `influxgraph.classes.tree.`**`NodeTreeIndex`**

Node tree index class with graphite glob searches per sub-part of a query

**clear**()

Clear tree index

**static from_array**(*model*)

Load tree index from array

**static from_file**(*file_h*)

Load tree index from file handle

**insert**(*metric_path*)

Insert metric path into tree index

**insert_split_path**(*paths*)

Insert already split path into tree index

**query**(*query*)

Return nodes matching Graphite glob pattern query

**search**(*node*, *split_query*, *split_path*)
Return matching children for each query part in split query starting from given node

**to_array**()
Return array representation of tree index

# CHAPTER 3

## Graphite template related functions

Graphite template parsing functions per InfluxDB's Graphite service template syntax

**exception** influxgraph.templates.**InvalidTemplateError**
Raised on Graphite template configuration validation errors

**exception** influxgraph.templates.**TemplateMatchError**
Raised on errors matching template with path

**class** influxgraph.templates.**TemplateFilter**(*pattern*)
Filter metric paths on template pattern

**match**(*path*)
Check if path matches template pattern

**Parameters** **path** (*str*) – Graphite path to check

**Return type** bool

**match_split_path**(*split_path*)
Go through split sub-paths and pattern's sub-paths and check if pattern matches all sub-paths

**Parameters** **split_path** (*list (str)*) – Graphite metric path split on separator

influxgraph.templates.**apply_template**(*metric_path_parts*, *template*, *default_tags*, *separator='.'*)
Apply template to metric path parts and return measurements, tags and field

**Raises** mod:*TemplateMatchError* on error matching template

influxgraph.templates.**get_series_with_tags**(*paths*, *all_fields*, *graphite_templates*, *separator='.'*)
Get list of metric paths from list of InfluxDB series with tags and configured graphite templates if any.

Without graphite template configuration tags are dropped and only the series name is used.

influxgraph.templates.**heapsort**(*iterable*)
Perform heap sort on iterable

**Parameters** **iterable** – Iterable with (index, value) tuple entries to sort

on index value. *index* must be integer, *value* can be anything :type iterable: *tupleiterator*

`influxgraph.templates.`**`parse_influxdb_graphite_templates`**(*templates*, *separator='.'*)

Parse InfluxDB template configuration and return parsed templates

**Parameters**

- **`templates`** (*`list(str)`*) – Template patterns to parse. Format is [filter] <template> [tag1=value1,tag2=value2]
- **`separator`** (*`str`*) – (Optional) Separator to use when storing greedy matched columns

**Raises** `InvalidTemplateError` on invalid template format used in any

template pattern

# CHAPTER 4

## Utility functions and classes

InfluxGraph utility functions

**class** `influxgraph.utils.`**`Query`**(*pattern*)

Graphite-API compatible query class

`influxgraph.utils.`**`calculate_interval`**(*start_time*, *end_time*, *deltas=None*)

Calculates wanted data series interval according to start and end times

Returns interval in seconds :param start_time: Start time in seconds from epoch :param end_time: End time in seconds from epoch :type start_time: int :type end_time: int :param deltas: Delta configuration to use. Defaults hardcoded if no

configuration is provided

**Return type** int - *Interval in seconds*

`influxgraph.utils.`**`gen_memcache_key`**(*start_time*, *end_time*, *aggregation_func*, *paths*)

Generate memcache key to use to cache request data

`influxgraph.utils.`**`gen_memcache_pattern_key`**(*pattern*)

Generate memcache key from pattern

`influxgraph.utils.`**`get_aggregation_func`**(*path*, *aggregation_functions*)

Lookup aggregation function for path, if any. Defaults to 'mean'.

**Parameters**

- **`path`** (*`str`*) – Path to lookup
- **`aggregation_functions`** (*`dict(<pattern>: <compiled regex>)`*) – Aggregation function configuration

`influxgraph.utils.`**`get_retention_policy`**(*interval*, *retention_policies*)

Get appropriate retention policy for interval provided

**Parameters**

- **`interval`** (*`int`*) – Interval of query in seconds

- **retention_policies** (*dict(max time range of interval in seconds: retention policy name)*) – Retention policy configuration

**Return type** `str` or `None`

`influxgraph.utils.`**`make_memcache_client`**(*memcache_host*, *memcache_max_value=1*)

Make memcache client if given a memcache host or *None*

`influxgraph.utils.`**`parse_series`**(*series*, *fields*, *graphite_templates*, *separator='.'*)

Parses series and fields with/without graphite templates and returns built Index

**Parameters**

- **series** (*list(unicode str)*) – Series to load
- **fields** (*dict(measurement: [field1, field2, .])*) – Per measurement field keys from InfluxDB. May be *None*
- **graphite_templates** – Graphite templates to use to parse series

and fields. :type graphite_templates: list(tuple) as returned by *influxgraph.templates.parse_influxdb_graphite_templates*

**Return type** *influxgraph.classes.tree.NodeTreeIndex*

`influxgraph.utils.`**`read_influxdb_values`**(*influxdb_data*, *paths*, *measurement_data*)

Return metric path -> datapoints dict for values from InfluxDB data

CHAPTER 5

# Indices and tables

- genindex
- modindex
- search

# Python Module Index

## i

influxgraph.classes.finder, 3
influxgraph.classes.tree, 5
influxgraph.templates, 7
influxgraph.utils, 9

# Index


## A

apply_template() (in module influxgraph.templates), 7

## B

build_index() (influxgraph.classes.finder.InfluxDBFinder method), 3

## C

calculate_interval() (in module influxgraph.utils), 9
clear() (influxgraph.classes.tree.NodeTreeIndex method), 5

## F

fetch_multi() (influxgraph.classes.finder.InfluxDBFinder method), 3
find_nodes() (influxgraph.classes.finder.InfluxDBFinder method), 3
from_array() (influxgraph.classes.tree.Node static method), 5
from_array() (influxgraph.classes.tree.NodeTreeIndex static method), 5
from_file() (influxgraph.classes.tree.NodeTreeIndex static method), 5

## G

gen_memcache_key() (in module influxgraph.utils), 9
gen_memcache_pattern_key() (in module influxgraph.utils), 9
get_aggregation_func() (in module influxgraph.utils), 9
get_all_series() (influxgraph.classes.finder.InfluxDBFinder method), 3
get_all_series_list() (influxgraph.classes.finder.InfluxDBFinder method), 3
get_field_keys() (influxgraph.classes.finder.InfluxDBFinder method), 3
get_retention_policy() (in module influxgraph.utils), 9
get_series() (influxgraph.classes.finder.InfluxDBFinder method), 3
get_series_with_tags() (in module influxgraph.templates), 7

## H

heapsort() (in module influxgraph.templates), 7

## I

InfluxDBFinder (class in influxgraph.classes.finder), 3
influxgraph.classes.finder (module), 3
influxgraph.classes.tree (module), 5
influxgraph.templates (module), 7
influxgraph.utils (module), 9
insert() (influxgraph.classes.tree.Node method), 5
insert() (influxgraph.classes.tree.NodeTreeIndex method), 5
insert_split_path() (influxgraph.classes.tree.NodeTreeIndex method), 5
InvalidTemplateError, 7
is_leaf() (influxgraph.classes.tree.Node method), 5

## L

load_index() (influxgraph.classes.finder.InfluxDBFinder method), 4

## M

make_memcache_client() (in module influxgraph.utils), 10
match() (influxgraph.templates.TemplateFilter method), 7
match_split_path() (influxgraph.templates.TemplateFilter method), 7

## N

Node (class in influxgraph.classes.tree), 5
NodeTreeIndex (class in influxgraph.classes.tree), 5

## P

parse_influxdb_graphite_templates() (in module influxgraph.templates), 8
parse_series() (in module influxgraph.utils), 10

## Q

Query (class in influxgraph.utils), 9
query() (influxgraph.classes.tree.NodeTreeIndex method), 5

## R

read_influxdb_values() (in module influxgraph.utils), 10

## S

save_index() (influxgraph.classes.finder.InfluxDBFinder method), 4
search() (influxgraph.classes.tree.NodeTreeIndex method), 5

## T

TemplateFilter (class in influxgraph.templates), 7
TemplateMatchError, 7
to_array() (influxgraph.classes.tree.Node method), 5
to_array() (influxgraph.classes.tree.NodeTreeIndex method), 6