



What's new in Graphite 1.1

Denys Zhdanov
@deniszh
FOSDEM 2018

Who am I

Denys Zhdanov

System engineer @ eCG / Marktplaats.nl

Twitter / Github: [@deniszh](#)

[Sysadmin Ninja](#) 🍷 📈 🌟

Graphite co-maintainer 🤠

Data geek 🧐

Pythonista 🐍

Gopher 🦔



What? Graphite?

1. Stores Time-Series data
2. Renders it

<http://graphiteapp.org/>



A composite image showing the Graphite web interface and the cover of the book "Monitoring with Graphite". The top part is a screenshot of the Graphite web application. It features a navigation bar with "graphite" in a large font, and links for "Login", "Documentation", and "User Interface: Dashboard | flit (experimental) | events (experimental)". Below this is a "Tree" view on the left showing a directory structure of metrics, with "carbon.agents.ip-10-6-51-71-a.metricsReceived" selected. On the right is the "Graphite Composer" window, which displays a line graph showing a sharp drop in a metric value from approximately 38,000 to 18,000 between 4 PM and 8 PM. The bottom part of the image shows the cover of the book "Monitoring with Graphite" by Jason Dixon, published by O'Reilly. The cover features a detailed illustration of a falcon perched on a branch.



Why do we need Graphite in 2018?

Great ecosystem: 80+ 3rd party tools on [Tools page!](#)

- Collectors (for system and components): [collectd](#) / [diamond](#) / [telegraf](#) / [statsd](#)
- Dashboards: (mostly [Grafana](#) now but also 20+ others)
- Monitoring/Alerting: [Maira](#) / [Seyren](#) / [Cabot](#) / [Grafana](#) etc.
- Compatible components:
 - relay/proxy: [carbon-c-relay](#) / [carbon-relay-ng](#) / [grafsy](#) / [gruffalo](#)
 - storage: [BigGraphite](#) / [metricatank](#) / [go-carbon](#) / [carbon-clickhouse](#) / [graphouse](#)
 - rendering: [carbon-api](#) / [graphite-api](#)



The good, the bad and ...

Pros:

- simple metric format

`<name> <value> <timestamp>\n`

- push model
- rich ecosystem

Cons:

- no tags!
- speed (use pypy or alternative backends)
- scalability (use alternative backends)
- hard to install (docker installation)



Tags, tags, tags AKA Dimensions

- `metric.name;tag1=value1;tag2=value2` - compatible with Carbon format and Graphite tooling.
- Writing path:
Normalization -> `_tagged/xxxx/yyyy/metric;tag=value.wsp` -> update TagDB
- TagDB and Tag API documentation -
<https://graphite.readthedocs.io/en/latest/tags.html>
Redis or Django-supported databases (SQLite / MySQL / Postgres)



Tags

Querying:

```
seriesByTag('tag=value')
```

```
seriesByTag('name=~cpu\..*', 'tag1!=value1') - PCRE regex is supported!
```

groupByTags - **same as** groupByNode

```
seriesByTag('name=disk.used', 'server=~web.*') | groupByTags('sumSeries', 'datacenter')
```

aliasByTags - **same as** aliasByNode

```
seriesByTag('name=disk.used', 'datacenter=dc1') | aliasByTags('server', 'name')
```



Tags: external integrations

Grafana: Tag support [PR#9230](#) (Grafana 4.7.0)

Auto-completion API for tags and values (Grafana 4.7.0)

Tag value regex filtering & multi valued template variables [PR#9911](#) (5.x)

Prometheus: Remote read / write

- [PR#3533](#) / [PR#3635](#) (based on example adapter)
- [Criteo's Graphite-Remote-Adapter](#) (including tags support now)
- Prometheus read/write api in Graphite (WIP) - [PR#2195](#) / [PR#735](#)



Python 3, finally

Python 3 finally supported in all 3 major components - whisper, carbon and graphite-web.

Should work with Python 2.7, 3.4, 3.5, 3.6 and PyPy.

Graphite-web is running on Django 1.11 LTS (supported up to 2020) .



Custom (user defined) functions

Pluggable functions, finally!

Put your function in `/opt/graphite/webapp/graphite/functions/custom`
or use `FUNCTION_PLUGINS = ['some.function_plugin',]`

Functions API

```
curl -s "http://graphite/functions?pretty=1"
```

(Support in Grafana planned - [PR#10505](#), will be in Grafana 5.0b1)

Custom function example: ASAP

ASAP: Prioritizing Attention via Time Series Smoothing <http://futuredata.stanford.edu/asap>
Please see details in <http://www.iamsysadmin.ninja/2018/01/asap-smoothing-in-graphite.html>





New clustering code

Pros: Local (whisper) and remote (cluster) calls are parallelized now.
`seriesByTag()` calls will propagate to cluster members.

`http://cluster/render?target=xxx&target=xxx&target=seriesByTag(...)`

Easier way to write 3rd party finders.

Cons: Beware of changes: some of 3rd party tools not ready yet, e.g.

- go-carbon (supported in master, but no tags yet)
- graphite-clickhouse (supported in master, but no tags yet)
- graphouse (doesn't support 1.1.x yet)



Pipe chaining for functions

```
alias(movingAverage(scaleToSeconds(sumSeries(stats_global.production.counter  
s.api.requests.*.count), 60), 30), 'api.avg')
```

is really

```
alias(  
    movingAverage(  
        scaleToSeconds(  
            sumSeries(  
                stats_global.production.counters.api.requests.*.count  
            )  
        , 60)  
    , 30)  
    , 'api.avg')
```



Pipe chaining for functions

```
alias(movingAverage(scaleToSeconds(sumSeries(stats_global.production.counters.api.requests.*.count), 60), 30), 'api.avg')
```

now can be written as

```
sumSeries(stats_global.production.counters.api.requests.*.count) | scaleToSeconds(60) | movingAverage(30) | alias('api.avg')
```

or even

```
stats_global.production.counters.api.requests.*.count | sumSeries() | scaleToSeconds(60) | movingAverage(30) | alias('api.avg')
```



Aggregation functions and xFilesFactor

Consistent aggregations' list:

average, median, sum, min, max, diff, stddev, count, range, multiply and last.

New aggregation functions

aggregate, aggregateWithWildcards, movingWindow, filterSeries, highest, lowest and sortBy.

Old functions are aliases for new ones e.g.

```
sumSeries(some.metric.*) -> aggregate(some.metric.*, 'sum')
```

xFilesFactor value to specify how many points in the window must be non-null for the output to be considered valid.

<https://grafana.com/blog/2016/03/03/25-graphite-grafana-and-statsd-gotchas/#runtime.consolidation>



What's next: beyond 1.1.x

Remove Django, but keep metric tree explorer and dashboard view as separate (optional) components

Fix for “new metric propagation delay” bug (workarounds are exists, e.g. Graphite-Clickhouse as cache)

Separate TagDB server (probably even in Go?)



Thank You!

Questions? issues? Contributions?

- <https://github.com/graphite-project>
- IRC: #graphite on [FreeNode](#)
- <https://answers.launchpad.net/graphite> / graphite@librelist.com