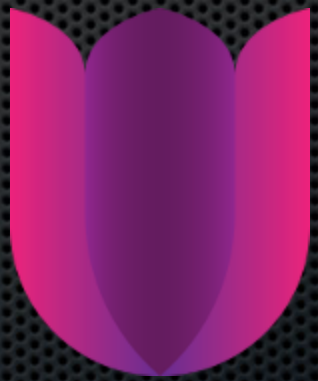
Grafana + Jsonnet

Julien Pivotto (@roidelapluie)



GrafanaCon 2018

```
{
    name: "Julien Pivotto",
    company: "Inuits",
    grafana: {
        first_issue: {
            id: $.grafana.first_pull.id + 1,
            date: "Apr 14, 2014",
        },
        first_pull: {
            // My first PR was somehow
            // related to dashboards as code
            // already
            id: 310,
            date: $.grafana.first_issue.date,
        },
    },
}
```

```
{
    "company": "Inuits",
    "grafana": {
        "first_issue": {
            "date": "Apr 14, 2014",
            "id": 311
        },
        "first_pull": {
            "date": "Apr 14, 2014",
            "id": 310
        }
    },
    "name": "Julien Pivotto"
}
```

Grafana Dashboards

- JSON
- Templates
- Annotations
- Panels
- Links

Grafana at scale

- Plenty of dashboards
- Consistency of templates
- Same links
- Same annotations
- Same panels

Panels Consistency

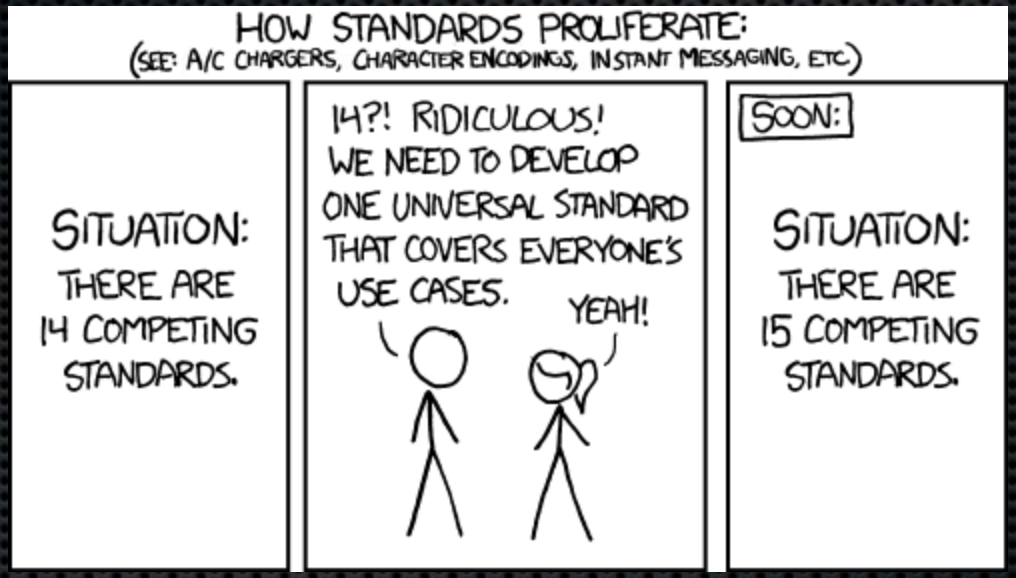
- Same color for given error code
- Same color for given server
- Same rules (stacked, lines, bars, width. datasource)

2 ways of solving that

- Spending way to much time building and correcting dashboards in the grafana UI
- Dashboards as code

Dashboards as Code

- https://github.com/uber/grafana-dash-gen
- https://github.com/weaveworks/grafanalib
- https://github.com/jakubplichta/grafana-dashboard-builder
- https://github.com/Showmax/grafana-dashboards-generator
- https://docs.openstack.org/infra/grafyaml
- https://docs.saltstack.com/en/latest/ref/states/all/salt.states.grafana4_dashboard.html



```
{
    Name: "jsonnet",
    "Open Source": true,
    License: "Apache License 2.0",
    Origin: "Google",
    URL: [
      "http://jsonnet.org/",
      "https://github.com/google/jsonnet",
      "https://github.com/google/go-jsonnet",
    ],
    Implementations: ["Golang", "C++"],
}
```

Jsonnet

> Jsonnet is a domain specific configuration language that helps you define JSON data.

http://jsonnet.org/

- Superset of JSON
- Functionnal language

Jsonnet, input

```
// Jsonnet Example
{
    person1: {
        name: "Alice",
        welcome: "Hello " + self.name + "!",
    },
    person2: self.person1 { name: "Bob" },
}
```

http://jsonnet.org/

Jsonnet, output

```
{
   "person1": {
      "name": "Alice",
      "welcome": "Hello Alice!"
   },
   "person2": {
      "name": "Bob",
      "welcome": "Hello Bob!"
   }
}
```

http://jsonnet.org/

Comments

Comments do not exist in JSON.

In Jsonnet:

```
// This.
/* And this. */
```

Rules for Humans

In JSON, commas are not allowed at the end of arrays

In Jsonnet:

```
[ 'commas', 'are', 'allowed',
  'at', 'the', 'end', 'of', 'arrays', ]
```

Simplicity

Json:

```
{"foo": "bar"}
```

Jsonnet:

```
{foo: "bar"}
```

Variables

```
local tool_name = "grafana";
local modules_total = 10;

{
    tool: tool_name,
    modules: modules_total,
}
```

Functions

```
{
  new(name, kind):: {
    oname: name,
    kind: kind,
    max: 10
  },
  foo: $.new("foo", "bar")
}
```

Imports

```
local grafana = import "grafana.libsonnet";
local dashboard = grafana.dashboard;

dashboard.new(
    "Device USE by slot",
    tags=["Technical", "Overview"],
)
```

stdlib

- Strings join
- Replace
- Maps
- Inserts
- Loops
- ...

Usage

```
jsonnet  foo.jsonnet > foo.json
```

Multi files:

```
jsonnet -m dashboards dashboards.jsonnet
```

Style enforcing

```
jsonnet fmt
```

Grafonnet

- Jsonnet library to build Grafana dashboards
- https://github.com/grafana/grafonnet-lib
- Same license & rules as Grafana
- We have tests! (Not enough docs yet)

Building dashboards with Grafonnet

```
local grafana = import 'grafonnet/grafana.libsonnet';
grafana.dashboard.new(
    'JVM',
    refresh='1m',
    time_from='now-1h',
    tags=['java']
)
.addTemplate(
    template.new(
        'env',
        'Prometheus',
        'label_values(jvm_threads_current, env)',
        label='Environment',
        refresh='time',
    )
)
```

```
// network_group.jsonnet
{
    "network.json":
        import "net/general.jsonnet",
    "by-family.json":
        import "net/by-family.jsonnet",
    "by-slot.json":
        import "net/by-slot.jsonnet",
    "by-subslot.json":
        import "net/by-subslot.jsonnet",
}
```

Example

- https://github.com/grafana/grafonnet-lib/blob/master/examples/jvm.jsonnet

Human readable values

Grafana expects:

```
{ sort: 0 }
```

In Grafonnet, you write:

```
{ sort: "decreasing" }
```

Going further

Put YOUR standards on top of Grafonnet:

```
{
    new(title, uid, tags=[], refresh="1m")::
        self +
        grafana.dashboard.new(
            title,
            uid=uid,
            refresh=refresh,
            tags=tags
        )
        .addTemplate(
            grafana.template.datasource(
                "PROMETHEUS_DS",
                "prometheus",
                "Prometheus MyCarenet 1",
                hide="value",
            )
        ),
}
```

Folders

```
std.mapWithKey(
    function(k, v) v {
        // Workaround for grafana/grafana#10895
        title: "Customer 1 - " + v.title,
        uid: "cust1" + v.uid,
    },
    {
        "jvm.json":
            local jvm = import "shared/jvm.libsonnet";
            jvm.new(datasource_regex="/.*Cust1.*/"),

    }
)
```

Reusing panels from existing dashboards

```
local existing="existing.json"

grafana.dashboard.new("My Dashboard")
.addPanels(existing.panels)
```

Standardize colors

```
graphPanel.new(
    "Frontend Error Rate",
    fill=8,
    legend_show=false,
    min=0,
    max=0.1,
    format="percentunit",
    show_xaxis=false,
    linewidth=0,
    decimals=2,
    datasource="-- Mixed --",
    nullPointMode="null as zero",
    legend_hideEmpty=true,
    stack=true,
    legend_hideZero=true,
)
+ colors.http
```

```
{
    http::
        {
            seriesOverrides: [
                {
                    alias: "/400$/",
                    color: "#629e51",
                },
            ]
        }
}
```

What if not implemented?

```
template.new(
    "instance",
    "$PROMETHEUS_DS",
    "label_values(up{job='$job'}, instance)",
    label="instance",
    multi=true,
    includeAll=true,
    current="all",
    refresh="load",
) + { sort: 1 }
```

Integration with Grafana

- Grafana 5 implements provisioning from files
- To be used with jonnet -m (multi)

Roadmap

- Listen to feedback
- Implement more feature
- Implement a layer abstraction (define dashboards without knowing internals) (opiniated)
- Find a way to document it

Thanks!

https://github.com/grafana/grafonnet-lib