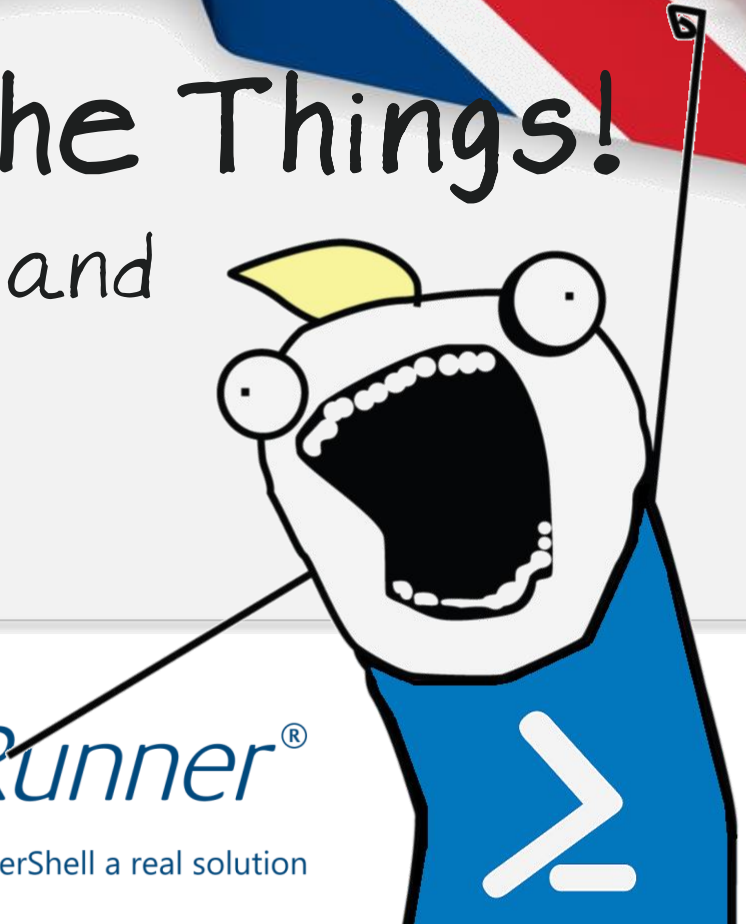
PSDAY.UK

2019

Measure All The Things!

With Influx, Grafana and PowerShell

Mark Wragg



IRONGATE WEALTH MANAGEMENT

ScriptRunner® Make PowerShell a real solution

About Me

- DevOps Engineer (contractor) for SolidSoft Reply.
- Ops background, transitioned to DevOps roles a few years ago.
- Presented a session about Pester at PSDay UK 2018.

Twitter: @markwragg

Blog: https://wragg.io

Github: https://github.com/markwragg

2019

About This Talk

Objectives:

- To introduce you to some interesting and free tools that you may not have seen/used before
- To inspire some creative use cases for dashboard / monitoring

Note:

- PowerShell can be used to gather monitoring data, but there may be better options for most simple metrics.
- The toolset described here probably shouldn't be your only approach to monitoring.
- This talk doesn't cover how to run these tools "in production".

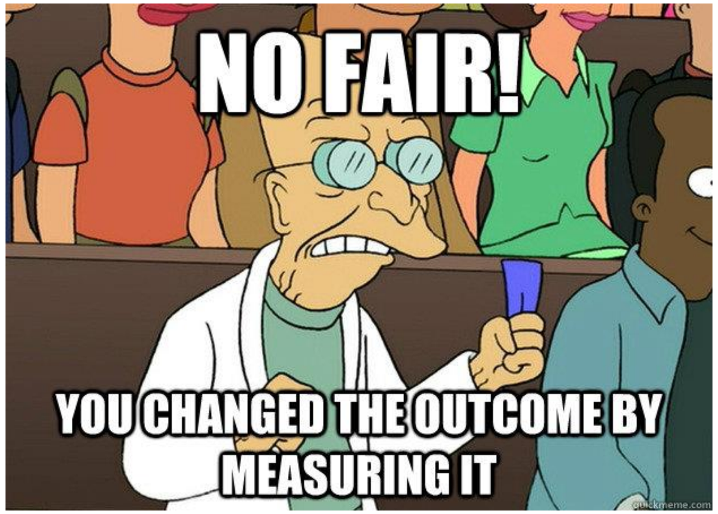
Why Monitoring is Important (for DevOps)

- DevOps is about going faster and enabling experimentation
- You can't do those safely without seeing how the changes you make are impacting the product
- Traditional monitoring focussed on availability. DevOps demands a focus on metrics and events.

Beware "The Observer Effect"

*"**the observer effect** is the theory that the mere **observation** of a phenomenon inevitably changes that phenomenon.*

This is often the result of instruments that, by necessity, alter the state of what they measure in some manner."



Select All The Tools

Influx

- A time series database.
- Alternatives: Graphite, Prometheus, OpenTSDB

Grafana

- Interactive dashboards / alerting.
- Alternatives: Graphite, Promteheus, Influx (The Chronograf component)

PowerShell

- Use scripts to collect metrics and transmit to Influx for visualisation with Grafana.
- Alternatives: Monitoring agents, Influx (Telegraf component)

Deploy All The Tools

- Both Influx and Grafana are simply executables with configuration files.
- Simple to install as Windows services using NSSM (Non-sucking Service Manager).
- Use my quick install script to install locally:
  https://github.com/markwragg/Presentations/blob/master/20190928_PSDayUK-2019/Code/0-InstallingTools.ps1
- Use Terraform to install in AWS or Azure:
  https://github.com/markwragg/Terraform-MetricStack/

Recording and Visualising Your First Metric

```
$InfluxConn = @{
  URI    = 'http://localhost:8086/write?db=metrics'
  Method = 'POST'
}
$Hostname = $env:ComputerName
$Region   = 'UKSouth'

While (1) {
  $CPU = ((Get-Counter '\Processor(_Total)\% Processor Time').CounterSamples |
    Where-Object { $_.InstanceName -eq '_total' }).CookedValue
  $Metric = "cpu_load,host=$Hostname,region=$Region value=$CPU"
  Invoke-RestMethod @InfluxConn -Body $Metric -Verbose
  Start-Sleep -Seconds 5
}
```

2019

Writing Metrics with the Influx Module

Available via the PowerShell Gallery:

```
Install-Module Influx
```

- Accepts input via hash tables
- Generates the Influx line protocol for you (handles escaping)
- Allows you to write multiple metrics with a single command

Usage Example:

```
$InfluxConn = @{
  Server   = 'http://localhost:8086'
  Database = 'metrics'
}

$Tags = @{
  Host   = $env:ComputerName
  Region = 'UKSouth'
}

$MemCounter = '\Memory\Available MBytes'
$CPUCounter = '\Processor(_Total)\% Processor Time'

While ($true) {
    $Metrics = @{
      Memory = (Get-Counter $MemCounter).CounterSamples.CookedValue
      CPU    = (Get-Counter $CPUCounter).CounterSamples.CookedValue
    }

    Write-Influx @InfluxConn -Measure 'Server' -Tags $Tags -Metrics $Metrics

    Start-Sleep -Seconds 5
}
```

Writing Metrics via UDP

- The **Write-Influx** cmdlet submits metrics via a TCP request.
- This could result in errors/delays in script execution if the endpoint is down or unreachable.
- Influx supports writing metrics via UDP (we configured it earlier).
- The Influx PowerShell module has a cmdlet for writing via UDP:

```
Write-InfluxUDP
```

TCP

UDP

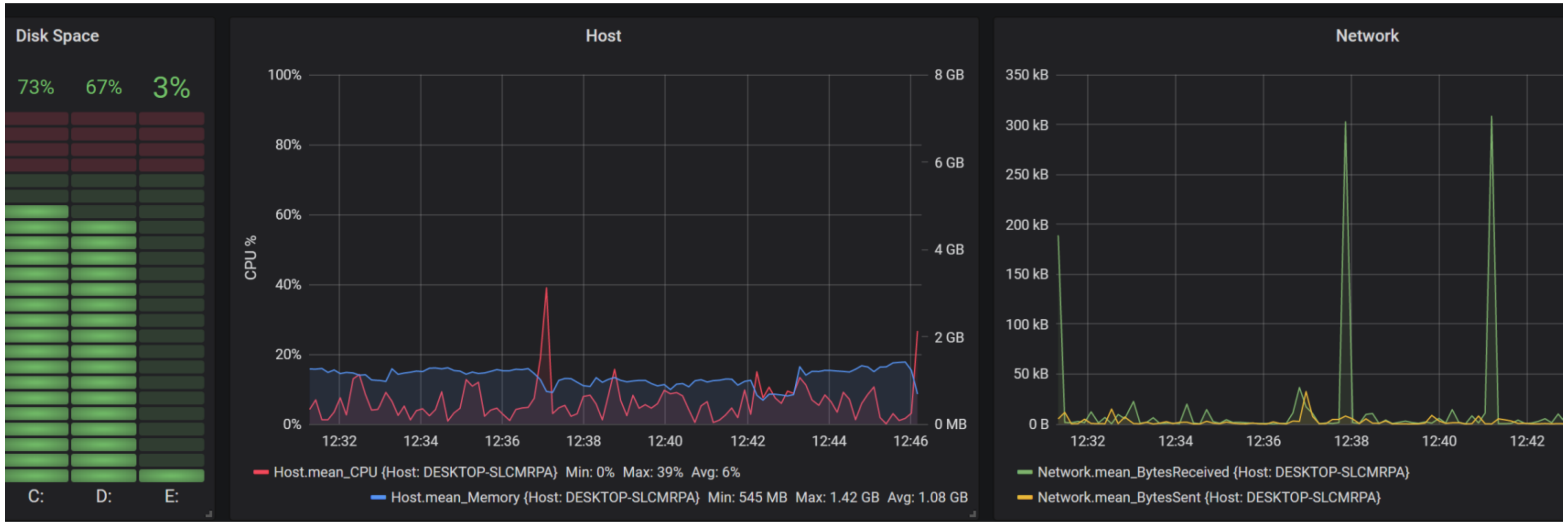
Monitoring with PowerShell

What should we monitor?

- **Operating System / Infrastructure** — Collect metrics for the health, performance and events that occur on your host systems and infrastructure.
- **Application** — Collect metrics from within the application, such as how long various internal logical tasks take to complete and where exceptions occur.
- **Business Logic** — Collect metrics that give the business insight in to the product, such as how many sales are made, the value of sales, and new user registrations.
- **Deployment Pipeline** — Collect metrics related to the deployment pipeline, such as how long deployments take to complete and the frequency and success of builds.

CPU / Memory / Disk / Network

- It is likely best to use built in agents to collect these kinds of metrics where available (particularly on Cloud Platforms)
- Failing that, on Windows this is easily done via WMI


Disk Space
73%
67%
3%
C:
D:
E:
Host
100%
80%
60%
40%
20%
0%
CPU %
8 GB
6 GB
4 GB
2 GB
0 MB
12:32
12:34
12:36
12:38
12:40
12:42
12:44
12:46
Host.mean_CPU {Host: DESKTOP-SLCMRPA} Min: 0% Max: 39% Avg: 6%
Host.mean_Memory {Host: DESKTOP-SLCMRPA} Min: 545 MB Max: 1.42 GB Avg: 1.08 GB
Network
350 kB
300 kB
250 kB
200 kB
150 kB
100 kB
50 kB
0 B
12:32
12:34
12:36
12:38
12:40
12:42
Network.mean_BytesReceived {Host: DESKTOP-SLCMRPA}
Network.mean_BytesSent {Host: DESKTOP-SLCMRPA}

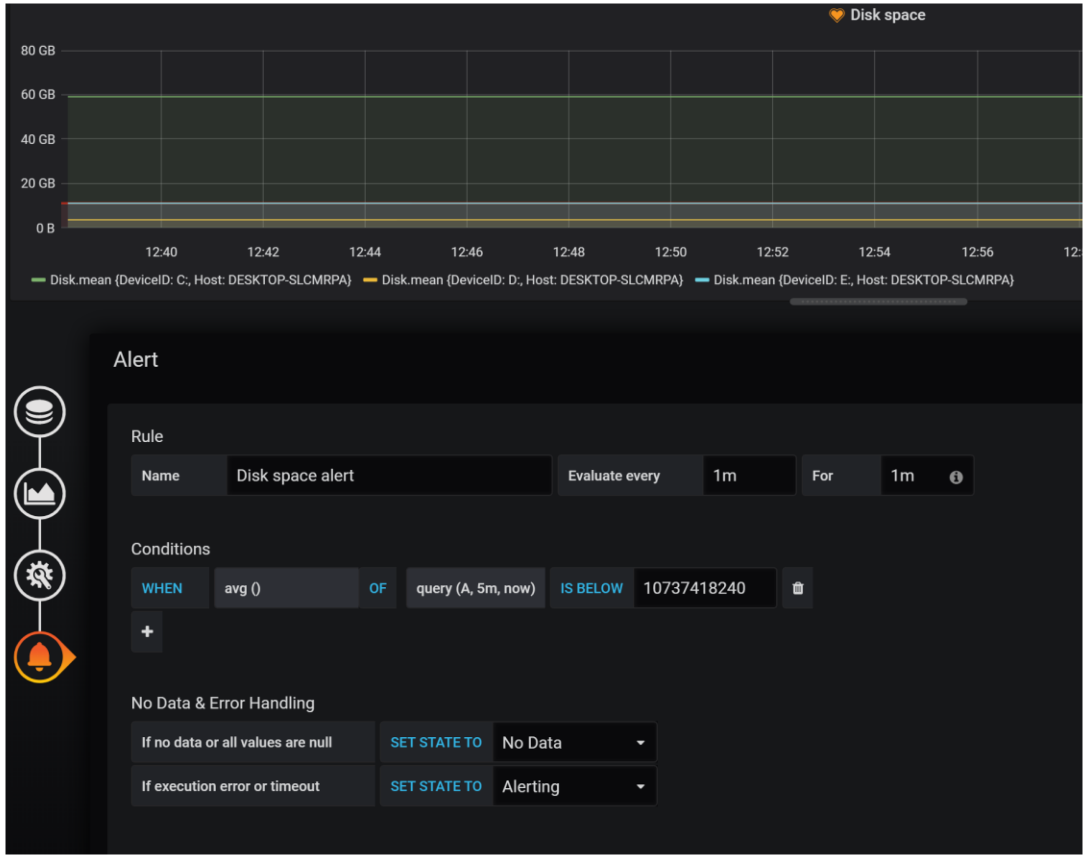
Alerting with Grafana

Grafana can be used as an alerting tool

- Edit a Graph
- Click on the Alert Tab > Create Alert
- Enter settings.

Note:

- Alerts can only be set on the graph visualisation.


Disk space
80 GB
60 GB
40 GB
20 GB
0 B
12:40
12:42
12:44
12:46
12:48
12:50
12:52
12:54
12:56
Disk.mean {DeviceID: C:, Host: DESKTOP-SLCMRPA}
Disk.mean {DeviceID: D:, Host: DESKTOP-SLCMRPA}
Disk.mean {DeviceID: E:, Host: DESKTOP-SLCMRPA}

Alert

Rule

Name: Disk space alert | Evaluate every: 1m | For: 1m

Conditions

WHEN avg () OF query (A, 5m, now) IS BELOW 10737418240

No Data & Error Handling

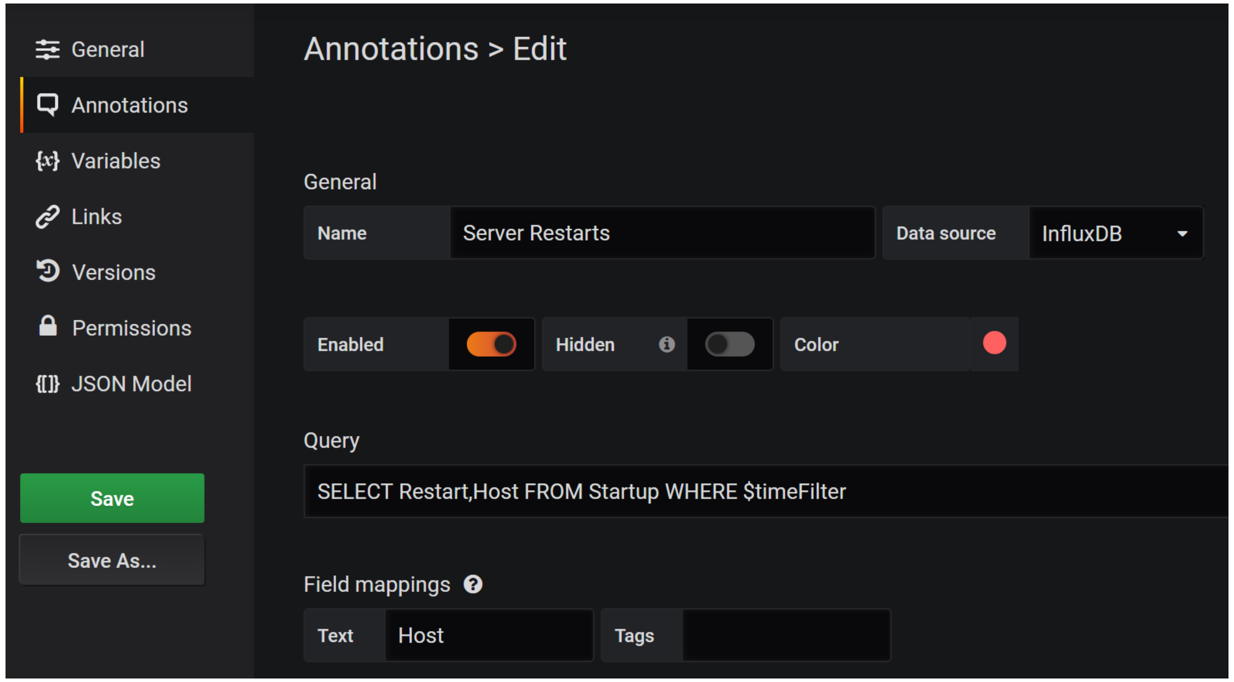
If no data or all values are null SET STATE TO No Data

If execution error or timeout SET STATE TO Alerting

Adding Annotations to Grafana

Annotations provide a useful way to overlay important events to graphs

- Annotations are added by alerts.
- Annotations can be added manually, as single points or ranges (CTRL+Click).
- Annotations can also be added automatically via a query of one or more measures →



Annotations > Edit

General

Name	Server Restarts	Data source	InfluxDB
Enabled	on	Hidden	off
Color	●		

Query

```
SELECT Restart,Host FROM Startup WHERE $timeFilter
```

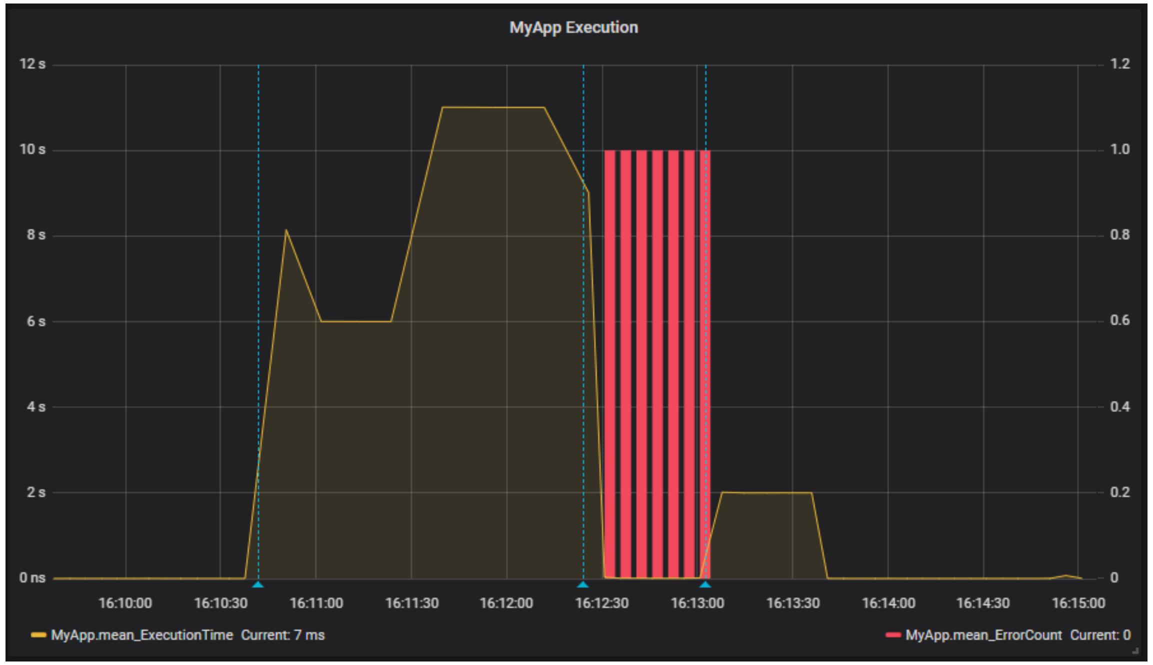
Field mappings

Text	Host	Tags	

Measuring Application Performance

Using Grafana you can visualise deployments alongside application performance metrics

- We have a fictious app called MyApp.
- We can use PowerShell and **Write-InfluxUDP** to track its execution time and errors.
- We track when app deployments occur as a metric.



Summary

- These tools are free, easy to use and can provide immediate value.
- The key is to make the impact of work visible.
- Provide monitoring/dashboarding as a service to the business – Empower stakeholders to build and manage their own dashboards with metrics that are useful to them.
- Build what works/is useful to you.
- Take a microservices approach to what and how you monitor.
- Consider what metrics are useful when combined.

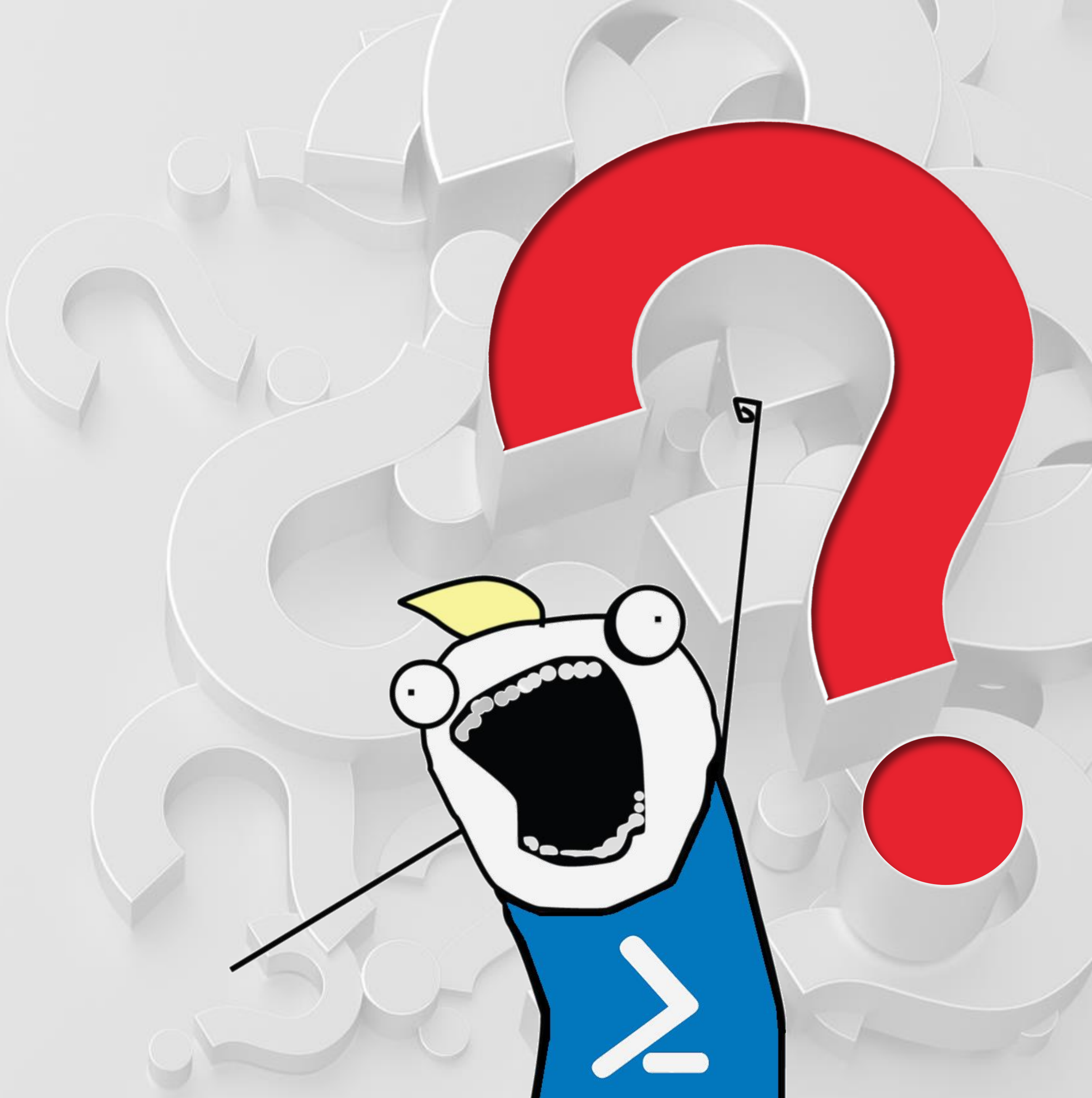
Influx: https://www.influxdata.com/

Grafana: https://grafana.com/

Code Examples: https://github.com/markwragg/Presentations/20190928_PSDayUK-2019/

PSDAY.UK

2019



PSDAY.UK
2019

Thank you

IRONGATE
WEALTH MANAGEMENT

ScriptRunner®
Make PowerShell a real solution

PSDAY.UK

2019