p3exporter Documentation

codeaffen

Mar 22, 2021

USER DOCUMENTATION

CHAPTER

ONE

PYTHON PROGRAMMABLE PROMETHEUS EXPORTER

PyPI version Codacy Badge Documentation Status

This repo should help any DevOps to quickstart its Prometheus exporter development base on Python. It is a POC which shows how different packages and libraries can be put together to create your own exporter for Prometheus.

The project is based on Flask as web framework and Prometheus python-client. We also provide a simple Dockerfile to enable you to build a Docker container image for your exporter as well as let it run as a simple program.

1.1 install and run locally

To install the `p3exporter` package you simply run the following command inside the project directory:

```
$ pip install -e .
Obtaining file:///home/nero/Development/p3exporter
...
Installing collected packages: p3exporter
  Running setup.py develop for p3exporter
Successfully installed p3exporter
```

From now you can run it with:

```
$ p3exporter
INFO:root:Start exporter, listen on 5876
```

1.1.1 available command line options

Do determine the available command line options you can call the online help:

```
$ p3exporter --help
usage: p3exporter [-h] [-c CONFIG] [-p PORT]

Python programmable Prometheus exporter.

optional arguments:
  -h, --help            show this help message and exit
  -c CONFIG, --config CONFIG
                        path to configuration file.
  -p PORT, --port PORT  exporter exposed port
```

1.2 build image and run in docker

To let the exporter run in docker you have to do the following:

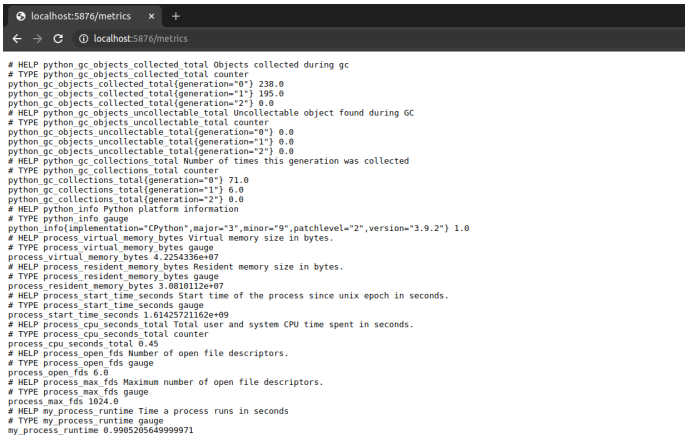
```
$ docker build -t codeaffen/p3exporter .
Sending build context to Docker daemon  181.8kB
...
Successfully built a6bdf60489f5
Successfully tagged codeaffen/p3exporter:latest
```

Now you can start the container. Don't forget to expose the port to your network.

```
$ docker run -d -p 5876:5876 --name p3e codeaffen/p3exporter
03e287d50cce595cec6ee66d75a663a094ba7688c761303e7e1e9ad39bde695c
```

1.3 access exporter

If your exporter run either as local program or as docker container you can access it with your webbrowser.



metrics in browser

CHAPTER

TWO

CHANGELOG

All notable changes to this project will be documented in this file. This project adheres to Semantic Versioning and Keep a Changelog.

2.1 Unreleased

2.1.1 New

2.1.2 Changes

2.1.3 Fixes

2.1.4 Breaks

2.2 1.0.0 - (2021-03-22)

2.2.1 New

- now it is simplier to add new collectors. You have to simply follow the naming convention
- add loadavg collector as a real life example

2.2.2 Breaks

- change load and registration behavior for collectors

2.3 0.1.0 - (2021-03-04)

2.3.1 Changes

- move collector to sub module

2.3.2 Fixes

- signal handling print now clean log messages instead of exceptions

CHAPTER

THREE

P3EXPORTER

3.1 p3exporter package

Init methods for p3exporter package.

p3exporter.**main**()
Start the application.

p3exporter.**shutdown**()
Shutdown the app in a clean way.

p3exporter.**signal_handler**(*signum*, *frame*)
Will be called if a signal was catched.

3.1.1 Subpackages

p3exporter.collector package

Entry point for collector sub module.

class p3exporter.collector.**Collector**(*config:* p3exporter.collector.CollectorConfig)
Bases: object

Base class to load collectors.

All collectors have to be placed inside the directory *collector*. You have to follow the naming convention:

1. Place the collector code in a <name>.py file (e.g. *my.py*)
2. Within the file <name>.py` a class <Name>Collector (e.g. *MyController*) needs to be defined. This is the main collector class which will be imported, instantiate and registered automatically.

class p3exporter.collector.**CollectorConfig**(***kwargs*)
Bases: object

Class that provide all the logic needed for configuration handling.

Submodules

p3exporter.collector.loadavg module

class p3exporter.collector.loadavg.**LoadavgCollector**(*config:* p3exporter.collector.CollectorConfig)

Bases: object

collect()

Collect load avg for 1, 5 and 15 minutes interval.

Returns three gauge metrics. One for each load.

p3exporter.collector.my module

class p3exporter.collector.my.**MyCollector**(*config:* p3exporter.collector.CollectorConfig)

Bases: object

A sample collector.

It does not really do much. It only runs a method and return the time it runs as a gauge metric.

collect()

Collect the metrics.

3.1.2 Submodules

3.1.3 p3exporter.web module

Web module provide all parts to create the web app.

p3exporter.web.**create_app**(*config:* p3exporter.collector.CollectorConfig)

Create the web app.

This Function creates the flask app and dispatch metrics endpoint to prometheus wsgi app.

Parameters **config** (CollectorConfig) – A configuration object with data for template rendering.

Returns Created web app object.

Return type DispatcherMiddleware

CHAPTER

FOUR

HOW TO CONTRIBUTE TO P3EXPORTER

4.1 Did you found a bug

- Make sure the bug is not already opened by another user.
- If you can't find an open issue which reflects your observed problem go ahead an open a new bug.
- Provide as much information as mentioned in the bug report template.

4.2 Did you wrote a patch for an open bug

- Open new pull request containing the patch.
- Provide a clear description which describes the problem and the solution. Link the existing bug to the PR.

4.3 Do you want to add a new feature

- Make sure there isn't already a feature request.
- If you can't find an open feature request which describe your feature idea or parts of it feel free to open a new feature request.
- Suggest your feature idea within the created feature request.
- Provide as much discribtion as possible to enable others to have a good understanding of what you are doing.
- Point out that you want to start to work on the new feature

4.4 Do you wnat to contribute to documentation

- Write you documentation change.
- Open a PR with your change.
- Discuss with the team about your changes.

4.5 Thank you for any contribution

We will thank you for heed the contribution guidelines and we encourage you to contribute and join the team.

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

p

INDEX