DORA Documentation

Release 0.1.0

Caian R. Ertl

Dec 25, 2020

Contents

1 Links & References **3**

2 Documentation Contents **5**

2.1 Usage 5

3 Indices and tables **7**

DNS **O**ver **R**EST **A**PI

DORA is a web application that provides a simple API for DNS querying through a REST architecture. It aims to be a consumable API that's easy to digest and easy to deploy in cloud-based solutions, such as Google App Engine.

DORA mainly relies on `dnspython` toolkit and the `flask` microframework.

CHAPTER 1

Links & References

- DORA on GitHub
- Flask microframework
- dnspython toolkit

CHAPTER 2

Documentation Contents

2.1 Usage

You can deal with DORA in two different ways: as a command-line interface (CLI) or as a RESTful Web API. If you're planning to deploy DORA on Heroku, Google App Engine or other PaaS, it is very unlikely that you ever touch the CLI.

The CLI was created with one simple intent in mind: provide an easy way to deal with DORA in IaaS services (like Google Compute Engine or Amazon EC2) or on your own machine. If you just want to try it out, you can easily install DORA and its core dependencies and run locally with a simple command like `dora start`.

The Web API provides the main functionality of the service, which is the capability of doing DNS queries over a REST architecture. Once the service is already deployed and ready to be consumed, you, your service/app/script or your customer can start querying with HTTP requests like `GET endpoint/dora/mx/domain.com`.

2.1.1 Command-line interface

The CLI will expect **at least one action or flag**. If none is provided, DORA will warns the user and exit with *code 1*:

```
user@machine:~$ dora

DORA: missing operand.
Try 'dora --help' for more information.
```

When providing the `--help` flag, this is what the user should see:

```
user@machine:~$ dora --help

usage: dora [-h] [-v] [--copyright] {start} ...

DORA's command-line interface.

DORA is a web application that provides a simple
```

(continues on next page)

(continued from previous page)

```
API for DNS querying through a REST architecture.

positional arguments:
  {start}        DORA commands
    start        starts DORA's service

optional arguments:
  -h, --help     show this help message and exit
  -v, --version  show the application version and exit
  --copyright    show the copyright information and exit

This is a Free and Open-Source Software (FOSS).
Licensed under the MIT License.

Project page: <https://github.com/caianrais/dora>
```

Subcommand structure

DORA tries to establish a git-like interface, where the top-level command is `dora` itself and the subcommands are treated like actions. In the *positional arguments* section of the help message, a list of subcommands will be shown.

The `start` action (which is a subcommand of DORA), for example, will provide a completely different output to the user when binded with the `--help` flag:

```
user@machine:~$ dora start --help

optional arguments:
-h, --help            show this help message and exit
-p PORT, --port PORT  sets the port number the application will listen to
                      default value: 80
-d, --debug           enable debug mode
```

Running locally

Provided that you've successfully installed DORA on your system, just use the `start` subcommand. On the example below, the TCP port number was specified along with debug mode:

```
user@machine:~$ dora start --port 8080 --debug

* Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 290-185-XXX
```

Warning: It is important to notice that TCP ports below 1000 are generally protected by the system and cannot be used without superuser privilege. For testing or development, it is best to use a port above 4000, like the 8080.

2.1.2 Web API

CHAPTER 3

Indices and tables

- genindex
- modindex
- search