page-monitor Documentation

Release

MarcDufresne

Mar 27, 2018

Contents:

1 Getting Started **3**

1.1 Installation . . . 3

1.2 Usage . . . 3

2 Configuration **5**

2.1 Config options . . . 5

3 Indices and tables **9**

page-monitor is an app that monitors web pages for changes, with optional conditions, and is able to execute actions based on changes, like send an email or a Telegram message.

It will also **fully support JavaScript** rendering by using Chromium in the background once `requests-html` releases an Async compatible version.

CHAPTER 1

Getting Started

1.1 Installation

You will need **Python 3.6+** to run this app.

An easy way to install Python 3.6 is to use pyenv (with pyenv-installer)

Install with pip

1. In a Python 3.6+ environment, run `pip install page-monitor`
2. Create a configuration file, see *Configuration*
3. Run the app according to *Usage*

Clone locally

1. Clone this repo
2. Create a configuration file, see *Configuration*
3. Install the contents of `requirements.txt` into a Python 3.6+ environment
4. Run the app according to *Usage*

1.2 Usage

Installed with pip

```
page_monitor <tasks_file>
```

Cloned locally

```
python page_monitor/monitor.py <tasks_file>
```

CHAPTER 2

Configuration

To configure tasks simply create a YAML file containing your tasks definitions. Here's an example:

```
tasks:
  - name: A Page
    url: http://example.com/page_1.html
    css_selector: ".a-class"
    first_only: True
    interval: 60
    actions:
      - type: email
        params:
          email_to: recipient@domain.com
    conditions:
      - type: text
        cond: has
        rule: "some text"
    condition_logic: and
redis: 127.0.0.1:6379
mailgun:
  domain: mail.domain.com
  api_key: key-aaa111bbb222ccc333
```

2.1 Config options

Here's a definition of the config structure with available options and default.

- **`tasks` *(list)*: List of tasks**
  - `name` *(string)*: Name for your task, useful for notifications, defaults to `url` if not specified.
  - `url` *(string)* **Required**: URL of the page to monitor.
  - `css_selector` *(string)* **Required**: Part of the page to monitor.

  - `first_only` *(boolean)*: If multiple elements match the `css_selector`. This will only process the first matched element. Defaults to: `False`.
  - `interval` *(integer)*: Interval in seconds for checks, minimum of 5 seconds, defaults to 1 hour (3600 seconds).
  - `render` *(bool)*: Whether to use Chromium to render JavaScript or not, defaults to `False`.
  - `actions` *(list)*: List of actions, see below for details.
  - `conditions` *(list)*: List of conditions, see below for details. If no conditions are specified actions will be triggered by any content change.
  - `condition_logic` *(string)*: Logic to apply to conditions, choices are `and` or `or`, defaults to `and`.

To run Page Monitor you will also need a Redis server, you can specify a connection string in a `host:port` format in a top level `redis` key. The `redis` config default to `127.0.0.1:6379`, so it can be omitted if using the default Redis config.

To send emails you will also need to specify a SMTP or Mailgun config. To do this, specify a top-level `mailgun` or `smtp` config with the following inside:

- **`mailgun`:**
  - `domain` *(string)* **Required**: Your Mailgun domain name.
  - `api_key` *(string)* **Required**: Your Mailgun API key, starts with `key-`.
  - `from_email` *(string)* **Required**: Email address that will be displayed as `From` in notification emails.
  - `from_name` *(string)*: The sender name that will be displayed in notification emails. Defaults to `from_email`.
- **`smtp`:**
  - `host` *(string)*: SMTP server host. Defaults to `smtp.gmail.com`.
  - `port` *(int)*: SMTP server port. Defaults to `587`.
  - `username` *(string)* **Required**: Username for SMTP server.
  - `password` *(string)* **Required**: Password for SMTP server.
  - `tls` *(bool)*: Whether to use TLS for server connection. Defaults to `True`.
  - `from_email` *(string)*: Email address that will be displayed as `From` in notification emails. Can be omitted if using GMail's SMTP server as it will rewrite it anyways.

2.1.1 Actions

Each item in the action list must first define what type of action it is, and contain any required or optional parameters for this specific action.

It should look like this:

```
type: action
params:
  param_1: value
  param_2: value
```

Supported Actions

- **`email`: Send an email with content diff when content has changed and conditions are met.**

  - `email_to` *(string)* **Required**: Recipient of the notification email.

- **`telegram`: Send a Telegram message when changes are detected and conditions are met.**
  - `chat_id` *(string)* **Required**: ID of the chat messages should be sent to.
  - `token` *(string)* **Required**: Your Telegram bot token, see Telegram Docs for details.

2.1.2 Conditions

Each condition has 3 attributes:

- **`type` Required: Defines what will be evaluated, should be one of the following:**
  - `text`: Check will be made on full current content.
  - `added_text`: Check will be made only on added text.
  - `removed_text`: Check will be made only on text that was removed.
- **`cond` Required: Defines what check will occur, should be one of the following:**
  - `not_empty`: Checks that the text is not empty.
  - `has`: Checks that the text contains the specified `rule`.
  - `does_not_have`: Inverse of `has`.
  - `matches_regex`: Check that the text matches the regex specified in `rule`.
- `rule` *(string)*: Rule used to check the condition, should be a plain string if using `has` or `does_not_have`, or a valid Python regex if using `matches_regex` (You can use Regex 101 for help with that).

CHAPTER 3

Indices and tables

- genindex
- modindex
- search